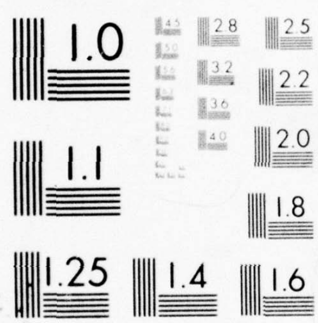
AD-A031 725 TEXAS UNIV AT AUSTIN DEFENSE RESEARCH LAB F/G 17/2
GENERATION OF A STATIONARY GAUSSIAN RANDOM PROCESS WITH A SPECI--ETC(U)
OCT 66 W A MATUSKA, G S INNIS NOBSR-93175
UNCLASSIFIED DRL-A-258 NL

1 OF 1
AD
A031725

END
DATE
FILMED
12-76



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

FENSE RESEARCH
LABORATORY

MOST Project #3

1

14 DRL-A-258

6 GENERATION OF A STATIONARY GAUSSIAN
RANDOM PROCESS WITH A SPECIFIED
POWER SPECTRAL DENSITY FUNCTION.

by

10 W. A. Matuska, Jr. and G. S. Innis

11 27 October 1966

Copy No. 3

12 72p.

9 DRL Acoustical Report No. 258

15
Resulting from research done under U. S. Naval
Ship Systems Command Contract NObsr-93175
Project Serial No. SF0010316, Task 8513 and Task 8515

DEFENSE RESEARCH LABORATORY
THE UNIVERSITY OF TEXAS
AUSTIN, TEXAS 78712

107 500
bpg

ABSTRACT

This paper, which is based on the technique published by Joel N. Franklin, gives a method for generating digitally a time series with a given power spectral density function. A computer program to carry out this method was written for the Control Data Corporation 3200 computer, and the program was tried on some specific example problems. Then for each of these examples, the power spectral density function of the generated time series was compared with the specified, theoretical power spectral density function.

TABLE OF CONTENTS

I. INTRODUCTION

A stationary Gaussian random process, $x(t)$, is called a time series if t represents discrete values of time. The collection of functions of time $x(t) = x_r(t)$ is called a random process if r is a random variable ranging over some measure space, R. The process is called Gaussian if, for every finite collection of times $t_1 < --- < t_n$, the random variables $x(t_1)$, ---, $x(t_n)$ have a multivariate Gaussian distribution. The process is called stationary if, for any increment Δt, the random variables $x(t_i+\Delta t)$ have the same joint distribution as the random variables $x(t_i)$.[1]

A common method for obtaining a time series is to record on magnetic tape the amplitude of a physical quantity, such as the pressure waves in a large body of water, the atmosphere, or the earth or the waves on the surface of a large body of water. This analog data can be thought of as a time series which can be either analyzed by means of an analog computer or converted into digital data and then analyzed by means of a digital computer. This method for obtaining a time series can be time consuming and the conditions which are necessary to obtain a time series with apecified statistical properties are often difficult to achieve; therefore, a method is desired for computing numerically a time series with given statistical properties.

This paper summarizes a technique, proposed by Joel N. Franklin[2], for computing a stationary Gaussian random process with a given power

spectral density function. It also contains a computer program for the Control Data 3200 computer which applies this technique in computing a specified stationary Gaussian random process along with four example problems which have been tried on this program.

A. Applications

The method given in this paper can be used to generate a stationary Gaussian statistical process with a specified power spectral density function which satisfies the conditions (1.8). Forms for the power spectral density function of many statistical processes can be found in the literature; for example, the general form for the power spectral density function of low frequency atmospheric turbulence is given by Bendat.[3] Problem four in Chapter III gives a specific example of such a power spectral density function.

As stated previously, a time series can be obtained by recording the amplitude of a statistical process, such as a pressure wave, at regular time intervals. However, when a time series is obtained in the field, the time series which is recorded on magnetic tape, called the total time series, is a combination of the time series obtained from the signal and the time series obtained from the noise. A signal is a detectable physical quantity or impulse by which messages or information can be transmitted, and noise is an unwanted detectable physical quantity or impulse which exists either by itself or in interference with a specified signal. Now if the power spectral density function of either of the three time series mentioned above is known and satisfies the conditions (1.8), a stationary Gaussian random process, which has the same power spectral density function as the time series obtained from the physical wave, can be generated digitally.

A problem of much interest is the study of detecting a known signal in various noise backgrounds. By the method given in this paper a time series can be generated which has the same power spectral density function as the time series obtained from a specified noise background. For example, suppose the time series of a noise-free signal of the infrasound (pressure waves having a frequency lower than about 16 cycles per second) produced by a thunderstorm is obtained. Then this time series can be viewed under any specified background conditions if the power spectral density function of the time series of the specified noise background is known. This can be accomplished by combining these two time series with the use of either the digital or analog facilities of a computer. The magnitude of the time series of the signal can be varied in relation to the magnitude of the time series of the noise, and by this method, the time series of the infrasound produced by a thunderstorm at a specified distance with a specified background can be simulated. The power spectral density functions of these total time series can be used in developing methods for detecting and tracking thunderstorms by giving examples of the power spectral densities of the infrasound produced by thunderstorms at various distances with various backgrounds.

The ability to obtain a time series for a given power spectral density function could also be useful in the advancement of the theory of optimum linear prediction and filtering. This theory is used in designing engineering systems which either project into the future by information obtained in the past or recover desired signals which have been distorted by random noise disturbances. These systems can be applied to communication, meteorological forecasting, and economic analysis. A specific example is the technique of combining two independently

obtained sets of related perturbed messages to obtain an improved estimate of the message.[4]

The last application to be mentioned is the study of smoothing techniques such as the hanning and hamming windows.[5] Here it is of interest to learn how the minor lobes of a power spectral density function are affected by the major lobe when these smoothing techniques are applied. This can be done by constructing time series for which the distances between the major and minor lobes of their power spectral density functions vary. The relative size of these lobes can also be varied along with the distance between them; therefore the effect of a smoothing technique on a given power spectral density function can be studied.

B. Summary of Method

The method under consideration for computing a stationary Gaussian random process is the method proposed by Joel N. Franklin.[6] In order to compute such a random process by this method, either the autocorrelation function, which is defined in terms of the expected value operator, or the power spectral density function must be known.

The expected value operator E is defined as

$$E[x_i] = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} x_i \tag{1.1}$$

if the limit exists. For a finite sequence x_1, the operation $\lim\limits_{N\to\infty}$ is neglected. Throughout this paper it is assumed that $E[x(t_i)]=0$ for each time series $x(t)$.

The autocorrelation function is defined as

$$R(t_1,t_2) = E[x(t_1)x(t_2)] \tag{1.2}$$

which is written as

$$R(\tau) = E[x(t)x(t-\tau)] \tag{1.3}$$

if $x(t)$ is a stationary process.

The power spectral density function for a stationary process is defined as

$$V(\omega) = \int_{-\infty}^{\infty} R(\tau)\, e^{-i\omega\tau} d\tau \tag{1.4}$$

if the integral exists.

The Fourier transform indicated by (1.4) can generally be inverted; therefore, if $V(\omega)$ is known and can be inverted, $R(\tau)$ can be found. Since it is more natural to work with the power spectral density function and since $V(\omega)$ and $R(\tau)$ are generally interchangeable, this method is designed to construct a stationary Gaussian random process from a given power spectral density function. Even though $V(\omega)$ and $R(\tau)$ are interchangeable, small perturbations in $V(\omega)$ may cause great changes in $R(\tau)$. This case is illustrated by example problem four in Chapter III.

Franklin attacks the problem by considering the linear transformation

$$x(t) = \int_{-\infty}^{t} g(t-s)\nu(s)ds \quad . \tag{1.5}$$

He then sets ν equal to the Dirac delta function; therefore $g(t)$ is the response of a filter to a delta function input. It is shown by Davenport and Root[7] that the power spectral density functions $V_x(\omega)$ and $V_\nu(\omega)$ of $x(t)$ and $\nu(t)$, respectively, are related by

$$V_x(\omega) = |\ G(\omega)\ |^2\ V_\nu\ (\omega) \quad , \tag{1.6}$$

where $G(\omega)$ is the Fourier transform of $g(t)$.

Now if a random process $\nu(t)$, known as white noise which has the property that $V_\nu(\omega)=1$, is obtained, (1.6) becomes

$$V(\omega) = V_x(\omega) = | G(\omega) |^2 \quad . \tag{1.7}$$

It is shown by Davenport and Root[8] that if

$$0<V(\omega)<\infty, V(\omega) = V(-\omega), \text{ and } V(\omega)\to 0 \text{ as } \omega\to\pm\infty, \tag{1.8}$$

and that if $V(\omega)$ can be represented as a rational function (i.e., the quotient of two polynomials) in ω, then for real ω, $V(\omega)$ can be represented by

$$V(\omega) = \left| \frac{P(i\omega)}{Q(i\omega)} \right|^2 \quad , \tag{1.9}$$

where P and Q are polynomials with real coefficients and of degree m and n, respectively, $m<n$, and the zeros of $Q(\xi)$ lie in the half-plane Re $\xi<0$. Now by (1.7) and (1.9), an obvious choice for $G(\omega)$ is

$$G(\omega) = \frac{P(i\omega)}{Q(i\omega)} \quad , \tag{1.10}$$

and if D is the differential operator, d/dt, we have

$$x(t) = \frac{P(D)}{Q(D)} \nu(t) \quad . \tag{1.11}$$

The specified time series $x(t)$ is found by first finding a steady-state solution to the differential equation

$$Q(D)\varphi(t) = \nu(t) \quad , \tag{1.12}$$

for all real t and then combining the derivatives of $\varphi(t)$ of order less than n by

$$x(t) = P(D)\varphi(t) \quad . \tag{1.13}$$

The first step in finding a solution $\varphi(t)$ to the differential equation (1.12) is to generate a completely equidistributed sequence ν_n which is also a "white sequence." This sequence is to be used as the digital representation of a white noise source.

A "white sequence" is defined to be a sequence for which

$$R(\tau) = 0 \text{ for } \tau \neq 0 \quad . \tag{1.14}$$

A sequence ν_n is said to be equidistributed by k's if k is a positive integer and if for every set of k intervals (a_i, b_i), i=1, ----, k,

where $0 \leq a_i < b_i \leq 1$,

$$\frac{1}{N} \sum_{\substack{a_i \leq \nu_{n+i} < b_i \\ (i=1,---,k) \\ n=1,---,N}} 1 \rightarrow \prod_{i=1}^{k} (b_i - a_i) \text{ as } N \rightarrow \infty \quad , \tag{1.15}$$

and a sequence equidistributed by k's for all k is called a completely equidistributed sequence.

Let the notation $\{\varphi\}$ denote the fractional part of φ. It is shown by Franklin that for almost every $\theta > 1$, $\{\theta^n\}$ is a completely equidistributed sequence[9]; furthermore he suggests that the sequence $\nu_n = \{\theta^n\}$, where θ is a transcendental number greater than one, be used as the required "white sequence." For example, if $\theta \approx 2.72$, $\nu_1 \approx \{2.72\} = .72$, $\nu_2 \approx \{2.72^2\} = .3984$, etc.

Because of computational difficulties which limit the length of ν_n for a given $\theta > 1$, several different sequences, $\nu_n^{(1)}, ---, \nu_n^{(s)}$, $(n = 0,1,2,---)$, may be needed for a single application. These sequences should be interlaced by

$$\nu_{ns+1} = \nu_n^{(1)}, \; \nu_{ns+2} = \nu_n^{(2)}, \; \cdots, \; \nu_{ns+s} = \nu_n^{(s)}, \; n = 0,1,2,-- \tag{1.16}$$

to form the required "white sequence." A few of these difficulties will be pointed out in Chapter II, Section A.

Now that an equidistributed sequence ν_n has been constructed on the range $0 \leq \nu < 1$, a sequence w_n, which will be called the w-sequence, of independent samples from the Gaussian distribution with mean 0 and variance 1 can be constructed by the method of Box and Muller.[10] This method is given by the two formulas,

$$\begin{aligned} w_{2n-1} &= (-2 \ln \nu_{2n-1})^{\frac{1}{2}} \cos 2\pi\nu_{2n} \\ w_{2n} &= (-2 \ln \nu_{2n-1})^{\frac{1}{2}} \sin 2\pi\nu_{2n} \text{ for } n=1,2,3,--- \quad . \end{aligned} \tag{1.17}$$

Now that the "white sequence," ν_n, and the w-sequence, w_n, have been defined, the next step in finding a solution to the differential equation (1.12) is to consider the given power spectral density function $V(\omega)$, which must be a rational function satisfying the conditions (1.8), and the given time increment Δt at which samples of the specified time series are to be calculated.

The equation (1.12) can be represented as

$$a_1\varphi^{(n)}(t) + a_2\varphi^{(n-1)}(t) + \text{---} + a_{n+1}\varphi(t) = \nu(t) \text{ for } -\infty<t<\infty \tag{1.18}$$

where $a_1=1$ and a_i, $1\leq i\leq n+1$, are the real coefficients of the polynomial Q. The vectors

$$z(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \\ \cdot \\ \cdot \\ \cdot \\ z_n(t) \end{bmatrix} = \begin{bmatrix} \varphi(t) \\ \varphi'(t) \\ \cdot \\ \cdot \\ \cdot \\ \varphi^{(n-1)}(t) \end{bmatrix} \cdot, \; t = 0, \Delta t, 2\Delta t, \text{----} \tag{1.19}$$

must be found in order to compute the specified time series by the equation (1.13). The vector z satisfies the differential equation

$$\frac{dz(t)}{dt} = A\, z(t) + f(t) \quad , -\infty < t < \infty \tag{1.20}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 \cdot & \cdot & 0 \\ 0 & 0 & 1 \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 \cdot & \cdot & 1 \\ -a_{n+1} & -a_n & -a_{n-1} & \cdot & -a_2 \end{bmatrix} \tag{1.21}$$

and

$$f(t) = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \\ v(t) \end{bmatrix} . \qquad (1.22)$$

In order to find $z(0)$, the solution vector $(m_1, m_2, --, m_n)$ must first be found to the set of n linear equations in n unknowns given by

$$(-1)^{k-1} \sum_{\frac{k}{2} + 1 \leq q \leq \frac{(n+k-1)}{2} + 1} (-1)^{q-1} a_{n-2(q-1)+k} m_q = \begin{cases} 0, & k=1, --, n-1 \\ 1/2, & k=n \end{cases} \qquad (1.23)$$

The elements of this solution vector are then placed into the matrix M by

$$m_{ij} = \begin{cases} 0, & i+j \text{ odd} \\ (-1)^{(j-i)/2} m_{(j+i)/2}, & i+j \text{ even} \end{cases} \qquad (1.24)$$

for $1 \leq i, j \leq n$.

The formulas (1.23) and (1.24) are proved by Franklin.[11] M is the positive definite moment matrix for the vector $z(t)$ for all time t, including $t = 0$. Since M is a positive definite, real, symmetric matrix, there exists a lower-triangular matrix T with positive elements on the main diagonal such that $M = T\,T.^*$ [12]

Now it becomes necessary to define the sequence of n-dimensional vectors,

$$w^{(0)} = \begin{bmatrix} w_1 \\ \cdot \\ \cdot \\ \cdot \\ w_n \end{bmatrix}, \quad w^{(1)} = \begin{bmatrix} w_{n+1} \\ \cdot \\ \cdot \\ \cdot \\ w_{2n} \end{bmatrix}, \quad w^{(2)} = \begin{bmatrix} w_{2n+1} \\ \cdot \\ \cdot \\ \cdot \\ w_{3n} \end{bmatrix}, \; --- \qquad (1.25)$$

whose elements are the elements of the w-sequence, and z(0) is determined by

$$z(0) = T\, w^{(0)} \quad . \tag{1.26}$$

Franklin shows that if $z(k\Delta t)$ is known, the following steps are needed to compute $z((k+1)\Delta t)$. $z(0)$ has been computed in equation (1.26); therefore, it is used as the starting value and $z((k+1)\Delta t)$ is found by induction. The first step in finding $z((k+1)\Delta t)$ is to compute the matrix $\exp(\Delta tA)$ with elements e_{ij}, $1 \leq i, j \leq n$.

It then becomes necessary to solve the set of simultaneous equations

$$\sum_{k=1}^{n} (a_{ik}\mu_{kj} + a_{jk}\mu_{ik}) = \begin{cases} \hat{b}_{ij} = e_{in}e_{jn}, & i<n \text{ or } j<n \\ \hat{b}_{nn} = e_{nn}^2 - 1, & i=n \text{ and } j=n \end{cases} \tag{1.27}$$

where μ_{ij} are the unknowns and a_{ij} are the elements of A for $1 \leq i,j \leq n$. The elements μ_{ij} are also the elements of the symmetric moment matrix M_r; therefore, $\mu_{ij} = \mu_{ji}$. It also can be seen that $\hat{b}_{ij} = \hat{b}_{ji}$; therefore, the n^2 equations in n^2 unknowns represented by (1.27) can be reduced to the $\frac{1}{2}n(n+1)$ equations in $\frac{1}{2}n(n+1)$ unknowns,

$$\sum_{k\leq j} a_{ik}\mu_{jk} + \sum_{k>j} a_{ik}\mu_{kj} + \sum_{k\leq i} a_{jk}\mu_{ik} + \sum_{k>i} a_{jk}\mu_{ki} = \hat{b}_{ij} \tag{1.28}$$

$$\text{for } 1 \leq j \leq i \leq n.$$

When the solution to (1.28) has been found, the positive definite, real, symmetric matrix M_r is known, and therefore a lower triangular matrix T_r can be found such that $M_r = T_r\, T_r^{*}$.

T_r, Δt, $\exp(\Delta tA)$, $z(0)$, and $w^{(k)}$, $k=1,2,---$, are now known, and the last step in evaluating $z((k+1)\Delta t)$ is carried out by

$$z((k+1)\Delta t) = \exp(\Delta tA)z(k\Delta t) + T_r w^{(k+1)} \tag{1.29}$$
$$\text{for } k = 0,1,2,--- \quad .$$

Now the solution $\varphi(t)$ to the equation (1.12) and its first (n-1) derivatives are known. Equation (1.13) becomes

$$x(t) = b_1 z_{m+1}(t) + b_2 z_m(t) + \cdots + b_{m+1} z_1(t), \quad t = 0, \Delta t, 2\Delta t, \cdots \qquad (1.30)$$

where $b_1 \neq 0$ and b_i, $1 \leq i \leq m+1$, are the real coefficients of the polynomial P and $z_i(t)$, $1 \leq i \leq n$, are the elements of the vector $z(t)$ as defined by (1.19). This completes the process since the specified time series $x(t)$ has been obtained.

II. COMPUTER PROGRAMS

The method for calculating a time series with specified power spectral density function, as presented in Chapter I, is divided into three separate programs. The first program, ISLASHO with subroutine RANDOM, generates a "white sequence" for a given θ and buffers this sequence out onto tape. The program CLLCNVRT, pronounced Call Convert, with subroutine CONVERT then converts the "white sequence" into the w-sequence by (1.17). This w-sequence is then buffered out onto a second tape which is the input tape to the program GAUSSIAN. GAUSSIAN then shapes the w-sequence into a time series with a given power spectral density function. The power spectral density function is defined in this program by the coefficients of the polynomials P and Q as defined by (1.9). This specified time series is buffered out onto a third tape. The same w-sequence can be shaped into any specified time series.

A. Generating a "White Sequence"

The purpose of the program ISLASHO is to generate a "white sequence" as described in Chapter I, Section B.

For the purpose of this paper, the transcendental number greater than one was chosen to be e = 2.71828183. The "white sequence", $\nu_n = \{e^n\}$ for $1 \leq n \leq 4266$, was then generated on the Control Data 3200 computer.

The following approach was taken to the problem of generating a "white sequence." The approach leads to a description of the method used.

For every $n > 1$, e^n can no longer be stored to full accuracy as as an eleven digit, floating point number; therefore, this is obviously not the approach to take.

It can be seen that about 1740 digits are necessary to store the integer part of e^{4000} since $\log(e^{4000}) \approx 1740$. Initially e is rounded to nine digits; then for the assurance that the fractional part of e^n, v_n, will not start repeating, e^n is saved with no round-off error. Since the present approximation of e contains eight digits to the right of the decimal point, $\{e^{4000}\}$ contains 32,000 digits, and therefore e^{4000} consists of approximately 33,740 digits. Now if these digits were stored in groups of threes as fixed point numbers, e^{4000} would require about 12,000 fixed point numbers for storage. The storage capacity of the Control Data 3200 is about 32,000 24-bit words; therefore, if two 12,000-word arrays are needed for the multiplication, the machine still has the capacity to handle the program.

The largest fixed point number that can be stored in a 24-bit word is $2^{23}-1 \approx 8,388,000$; therefore, all six-digit and most seven-digit integers can be stored in one 24-bit word.

Now the properties that the Control Data 3200 can handle two 12,000, 24-bit word arrays and all six-digit integers can be stored in a 24-bit word will be used in the program ISLASHO to raise any seven-, eight-, or nine-digit number to powers and save the fractional part. The two 12,000-word arrays, IX and IY, are used to store the products as three-digit, fixed point numbers. These products should be thought of as numbers represented in the base 1000. The notation used in equations (2.1), (2.4), and (2.7) is that of "long hand" multiplication.

The number which is to be raised to powers is read into ISLASHO as the three, three-digit, fixed point numbers, IX3, IX2, and IX1, with IX1 containing the three right-most digits. In the method used in ISLASHO, we first set IX(1) = IX1, IX(2) = IX2, IX(3) = IX3, and L = 3

and then start the following procedure.

In step one we have

$$\begin{array}{ccccc} & \mathrm{IX(L)}\ \cdots\cdot & \mathrm{IX(3)} & \mathrm{IX(2)} & \mathrm{IX(1)} \\ & & & & \mathrm{IX1} \\ \cline{2-5} \mathrm{IY(L+1)} & \mathrm{IY(L)}\ \cdots\cdot & \mathrm{IY(3)} & \mathrm{IY(2)} & \mathrm{IY(1)} \end{array} \quad . \tag{2.1}$$

IY(1) is computed by

$$\mathrm{IY(1)} = R\left(\frac{\mathrm{IX(1)} \times \mathrm{IX1}}{1000}\right) \quad ; \tag{2.2}$$

also let

$$\mathrm{ICARRY}_1 = I\left(\frac{\mathrm{IX(1)} \times \mathrm{IX1}}{1000}\right) \quad .$$

$R\left(\frac{a}{b}\right)$ is defined to be the remainder of a/b, and $I\left(\frac{a}{b}\right)$ is defined to be the integral number of times b divides a where a and b are integers.

Then for $2 \leq i \leq L$,

let $\mathrm{IY}(i) = R(T_i)$ and $\mathrm{ICARRY}_i = I(T_i)$

where

$$T_i = \left(\frac{\mathrm{IX}(i) \times \mathrm{IX1} + \mathrm{ICARRY}_{i-1}}{1000}\right) \quad . \tag{2.3}$$

To terminate this step, set $\mathrm{IY(L+1)} = \mathrm{ICARRY}_L$ and increase L by one.

In step two we have

$$\begin{array}{cccccc} & & \mathrm{IX(L-1)}\ \cdots & \mathrm{IX(3)} & \mathrm{IX(2)} & \mathrm{IX(1)} \\ & & & & \mathrm{IX\ 2} & \\ \hline \mathrm{IY(L+1)} & \mathrm{IY(L)} & \mathrm{IY(L-1)}\ \cdots & \mathrm{IY(3)} & \mathrm{IY(2)} & \mathrm{IY(1)} \end{array} \quad , \tag{2.4}$$

where IY(1) has the same value as in step one, L-1 has the same integer value as L in step one, and $\mathrm{IY(2)} = R(T_2)$ and $\mathrm{ICARRY}_2 = I(T_2)$ for

$$T_2 = \frac{\mathrm{IX(1)} \times \mathrm{IX2} + \overline{\mathrm{IY(2)}}}{1000} \quad . \tag{2.5}$$

Let $\overline{\mathrm{IY}(i)}$ denote IY(i) as computed in the previous step.

Let $\mathrm{IY}(i) = R(T_i)$ and $\mathrm{ICARRY}_i = I(T_i)$,

where

$$T_i = \left(\frac{\mathrm{IX}(i-1) \times \mathrm{IX2} + \overline{\mathrm{IY}(i)} + \mathrm{ICARRY}_{i-1}}{1000}\right) \text{ for } 3 \leq i \leq L \quad . \tag{2.6}$$

Step two is also terminated by setting $IY(L+1) = ICARRY_L$ and then increasing L by one.

In step three we have

$$\begin{array}{rrrrrrr} & & & IX(L-2) \cdots & IX(3) & IX(2) & IX(1) \\ & & & & IX3 & & \\ \hline IY(L+1) & IY(L) & IY(L-1) & IY(L-2) \cdots & IY(3) & IY(2) & IY(1) \end{array} \tag{2.7}$$

where IY(1) and IY(2) remain invariant from step two, $IY(3) = R(T_3)$, and $ICARRY_3 = I(T_3)$ where

$$T_3 = \left(\frac{IX(1) \times IX3 + \overline{IY(3)}}{1000} \right) \quad . \tag{2.8}$$

Let $IY(i) = R(T_i)$ and $ICARRY_i = I\ (T_i)$

where

$$T_i = \left(\frac{IX(i-2) \times IX3 + \overline{IY(i)} + ICARRY_{i-1}}{1000} \right) \tag{2.9}$$

for $4 \leq i \leq L$.

To terminate step three we set

$$IY(L+1) = \begin{cases} ICARRY_L, \text{ if } ICARRY_L \neq 0 \\ \text{not defined, if } ICARRY_L = 0 \end{cases} \quad , \tag{2.10}$$

$$\hat{L} = \begin{cases} L+1, \text{ if } ICARRY_L \neq 0 \\ L, \text{ if } ICARRY_L = 0 \end{cases} \quad , \tag{2.11}$$

and then $L = \hat{L}$.

It can be seen from equations (2.1) through (2.9) that no fixed point number calculated by the above method will be greater than $999^2 + 999 + 999 = 999{,}999$ which is a six-digit number and can therefore be handled by the Control Data 3200.

The IY array is then placed into the IX array and eight, nine, or ten correct digits (depending on where the theoretical decimal point is located in relation to the three digits of IX(i) which contains the theoretical decimal point) after the theoretical decimal point are placed into the floating point X array. The X array therefore contains numbers between zero and one; this is the desired "white sequence." The process, starting with step one and continuing through the placing of a new number

into the X array, is repeated until L is even and ISTOP $< L \leqq 12{,}000$ where ISTOP is an input parameter.

The entire IX array is placed on tape before ISLASHO terminates; therefore, this calculation can later be continued. However, in order to calculate many more than 4266 points in this "white sequence" by this method, ISLASHO would have to be modified such that part of the IX and IY arrays could be stored elsewhere than in the memory of the computer during the calculation of each point of X because these two arrays would soon out grow the storage capacity of the computer. Also the longer these arrays become, the time needed to calculate each point of X increases until this method would no longer be practical. For example, ν_n, $1 \leqq n \leqq 500$, was computed in about 45 seconds as compared with thirteen minutes needed to compute the 500 points, ν_n, $3501 \leqq n \leqq 4000$. It was noted that if t minutes were needed to compute a block of 500 points of ν_n, approximately $(t + 1.75)$ minutes were needed to compute the next 500 points of ν_n. It therefore may become necessary to generate several "white sequences," $\nu_n^{(1)}, \ldots \nu_n^{(s)}$, for different transcendental numbers and then interlace these sequences by the method (1.16) to form the desired "white sequence."

The following is the list of input parameters to the program ISLASHO. The numbers in parentheses were the numbers used to find $\{e^n\}$ for $1 \leq n \leq 4266$.

NO DEC	the number of digits to the right of the decimal point of the number which is to be raised to power. (8).
IX3, IX2, IX1	these three parameters each contain three digits of the number which is to be raised to powers. (2.71828183 was used; therefore, IX3 = 271, IX2 = 828, and IX1 = 183).
IF PRNT SQ = 1,	the X array is buffered out onto tape and printed in blocks of IBUF points,
= 0,	the X array is only buffered out on tape in blocks of IBUF points. (1).
IF PRNT ME = 1,	buffer out onto tape and print the entire IX array just before the program terminates,
= 0,	only buffer out onto tape the entire IX array (0).
ISTOP	the program terminates when L is even and ISTOP < L ≤ 12,000. ISTOP should never exceed six less than the dimension on the IX and IY arrays. (11990).
IBUF	the number of points of the "white sequence," X array, that should be handled in one block. IBUF should be an even number and not exceed the dimension of X which is 500 in the listing. Also IBUF ≤ 588 if the subroutine SOLUTION in GAUSSIAN is not to be changed.

The output tape has the following form:

IBUF

X(1) · · · · X(IBUF)

.

.

.

IBUF

X(1) · · · · X(IBUF)

ICOUNT

X(1) · · · · X(ICOUNT)

end-of-file

IDEC

L

IX(1) · · · · IX(L)

end-of-file .

ICOUNT — the number of points in the last block of the X array. ICOUNT ≦ IBUF.

I DEC — the number of digits in the decimal part of the last number in memory.

L — the number of three-digit numbers which compose the last number in memory.

IX(1) — IX(L) — the last number in memory put on tape as three-digit numbers. IX(1) is the right most and IX(L) is the the left most three digits of this number.

B. Constructing the w-sequence

The only purpose of CLLCNVRT is to convert a "white sequence" into its w-sequence by (1.17) and place this w-sequence onto tape. This program is almost trivial, but it simplifies the programming of the problem. It can be seen by (1.17) why it is desired that an even number of points of the "white sequence" be placed on tape in each block. If the number of points in a block is odd, the last point in that block will not be used by the subroutine CONVERT in forming the w-sequence.

The program CLLCNVRT has one input parameter, IF PRINT.

IF PRINT = 1, print and buffer out the w-sequence onto tape,

= 0, only buffer out the w-sequence onto tape.

The output tape has the following form:

I BUF

W(1) .... W(I BUF)

.

.

.

I BUF

W(1) ... W(I BUF)

end-of-file.

I BUF same as on input tape.

W(1) ... W(I BUF) a block of w-sequence points.

C. Constructing a Specified Time Series

The program GAUSSIAN shapes the input w-sequence into a time series $x(t)$ with a given power spectral density function and a given sampling time interval. The power spectral density function must be defined by the real numbers a_i, $1 \leq i \leq n + 1$, and b_i, $1 \leq i \leq m+1$ for

$$\nu(\omega) = \left|\frac{P(i\omega)}{Q(i\omega)}\right|^2 = \left|\frac{b_1(i\omega)^m + \cdots + b_m(i\omega) + b_{m+1}}{a_1(i\omega)^n + \cdots + a_n(i\omega) + a_{n+1}}\right|^2 \qquad (2.12)$$

where $m < n$, ω is real, and the zeros of $a_1\xi^n + \cdots + a_n\xi + a_{n+1}$ lie in the halfplane $\mathrm{Re}\ (\xi) < 0$. It should be noted that each of the n-dimensional vectors in (1.25) is used to construct one point of the specified time series; therefore, if the w-sequence consists of J points, the specified time series will contain J/n points.

The numbers a_i, $1 \leq i \leq n+1$, are read into the vector SA; the numbers b_i, $1 \leq i \leq m+1$, are read into the vector SB; and the time interval Δt is read into DT.

Subroutine FIND B places the numbers, a_i, $1 \leq i \leq n+1$, into a constant, $n \times n$ matrix B and constructs the n-dimensional constant vector $[0, \cdots, 0, \frac{1}{2}]$ by (1.23).

The subroutine GAUSS P then finds the solution vector $[m_1, \cdots, m_n]$ by means of Gaussian elimination with partial pivoting.[13] Since the determinant of the input matrix can be found by simply computing the product of the elements on the main diagonal and multiplying this product by the proper sign after the input matrix has been reduced to upper triangular form, the determinant of the input matrix is always calculated by GAUSS P and printed out by the program GAUSSIAN as a guide to the reliability of the solution vector.

The subroutine FIND CM places m_i, $1 \leq i \leq n$, into the positive definite moment matrix M whose elements, m_{ij}, $1 \leq i, j \leq n$, have been found by (1.24). Then as stated in Chapter I, Section B, a lower triangular matrix T must be found such that

$$M = T\,T.^{*} \tag{2.13}$$

Since M is a positive definite, real, symmetric matrix, the desired lower triangular matrix T can be found by Crout factorization. Let t_{ij} be the elements of T and m_{ij} be the elements of M. It is known that $t_{ij} = 0$ for $j > i$; therefore, only t_{ij} for $j \leq i$ need to be computed. Then by (2.13)

$$m_{ij} = \sum_{k=1}^{j} t_{ik}\, t_{jk} \tag{2.14}$$

can be obtained. It is then suggested that t_{ij}, for $j \leq i$, be computed in the following order:

$$ij = 11,\ 21,\ \cdots,\ n1;\ 22,\ 32,\ \cdots,\ n2;\ \cdots;\ nn.$$

By this order,

$$t_{11} = m_{11}^{\frac{1}{2}} \tag{2.15}$$

is computed first, and the other elements in the first column are given by

$$t_{i1} = t_{11}^{-1}\, m_{i1} \quad \text{for } i = 2, \cdots, n \quad . \tag{2.16}$$

If the preceding columns $k < j$ have been computed, the $j^{\underline{th}}$ diagonal element can be computed by

$$t_{ij} = \left(m_{ij} - \sum_{k=1}^{j-1} t_{jk}^{2} \right)^{\frac{1}{2}} \quad . \tag{2.17}$$

Now the elements below the diagonal can be computed by

$$t_{ij} = t_{jj}^{-1}\ m_{ij} - \sum_{k=1}^{j-1} t_{ik}\ t_{jk} \tag{2.18}$$

for $i = j+1, \cdots, n$ where $j < n$.[14] This method of Crout factorization is carried out in the subroutine T T STAR.

The matrix $\exp(\Delta tA)$, where A has the form as shown in (1.21) and is constructed by the subroutine MAKE A, is computed by the subroutine EXP ADT. This exponential matrix is defined by

$$\exp(\Delta tA) = I + \frac{\Delta t}{1!} A + \frac{\Delta t^2}{2!} A^2 + \cdots + \frac{\Delta t^k}{k!} A^k + \cdots \tag{2.19}$$

which is the same as

$$\begin{aligned} \exp(\Delta tA) = I &+ \left(\frac{\Delta t}{1}\right) A \left[I \right] + \left(\frac{\Delta t}{2}\right) A \left[\frac{\Delta t}{1!} A\right] \\ &+ \left(\frac{\Delta t}{3}\right) A\left[\frac{\Delta t^2}{2!} A^2\right] + \left(\frac{\Delta t}{4}\right) A\left[\frac{\Delta t^3}{3!} A^3\right] + \cdots \\ &\cdots + \left(\frac{\Delta t}{k}\right) A\left[\frac{\Delta t^{k-1}}{(k-1)!} A^{k-1}\right] + \cdots \end{aligned} \tag{2.20}$$

The method of summation given by (2.20), where the $(k-1)^{\underline{th}}$ term is multiplied by $(\Delta t/k)$ A and then added to the previous sum, is used to calculate $\exp(\Delta tA)$ in EXP A DT.

The summation terminates when the condition that $b_{ij} < \epsilon_{chk} = \max \{|a_{ni}| \times 10^{-12} \mid 1 \leq i \leq n$ and a_{ni} is an element of A$\}$ holds for every b_{ij}, $1 \leq i, j \leq n$, where b_{ij} are the elements of $B = \frac{\Delta t^k}{k} A^k$ for $k \geq 1$. The maximum of the above set was taken instead of the minimum so that the computer would not set ϵ_{chk} equal to some very small floating point numbers which should actually be zero. However, if all a_{ni} are approximately the same in magnitude, no problem should arise. If a_{ni} vary greatly in magnitude, it may become necessary to define a floating point zero and change the subroutine to check for the minimum value of $\{|a_{ni}| \mid 1 \leq i \leq n\}$.

Let $\hat{a}_{ij}$ be the elements of $\hat{A}$ where

$$\hat{A} = A{\times}B = A\left(\frac{\Delta t^k}{k!}\,A^k\right) = \left(\frac{\Delta t^k}{k!}\right) AA^k = \left(\frac{\Delta t^k}{k!}\,A^k\right) A = B{\times}A \quad . \tag{2.21}$$

Now since $A{\times}B = B{\times}A$, the matrix multiplication for (2.21) can be defined by

$$\begin{aligned} \hat{a}_{ij} &= b_{i+1,j}, \quad 1 \leqq i \leqq n-1,\ 1 \leqq j \leqq n \\ \hat{a}_{n1} &= b_{nn}\, a_{n1}, \end{aligned} \tag{2.22}$$

and $\quad \hat{a}_{n,j+1} = b_{nj} + b_{nn} a_{n,j+1}, \quad 1 \leqq j \leqq n-1 \quad .$

This is the method of matrix multiplication used in EXPADT.

Now subroutine BIGSET constructs the $\frac{1}{2}\,n(n+1)$ equations in the same number of unknowns by (1.28) where the $\hat{b}_{ij}$ for $1 \leqq j \leqq i \leqq n$ are defined by (1.27). The unknowns μ_{ij}, $1 \leqq j \leqq i \leqq n$, can be written in the form of an $n{\times}n$ lower triangular matrix L, and a one-to-one mapping f can be defined which maps the elements on and below the main diagonal of L onto an $\hat{n}$-dimensional vector v where $\hat{n} = \frac{1}{2}\,n(n+1)$. This vector is then the vector of unknowns associated with the $\hat{n} \times \hat{n}$ constant matrix defined by (1.28). After the value of v has been found by GAUSS P, an inverse mapping can be defined to take the elements of v back into L. L can then be made into the desired real symmetric matrix M_r by replacing the zeros above the main diagonal with the corresponding elements below the diagonal.

Let $f{:}L \rightarrow v$ be the mapping

$$f{:}\begin{pmatrix} \mu_{11} & 0 & \cdots & 0 \\ \mu_{21} & \mu_{22} & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 \\ \mu_{n1} & \mu_{n2} & \cdots & \mu_{nn} \end{pmatrix} \longrightarrow \tag{2.23}$$

$$(\mu_{11}, \mu_{21}, \cdots, \mu_{n1}; \mu_{22}, \cdots, \mu_{n2}; \mu_{33} \cdots; \mu_{nn})$$

$$= (v_1, v_2, \cdots, v_{\hat{n}}) \quad .$$

When μ_{ij} is under consideration, all n elements in the first j-1 columns plus the upper i-1 elements in the $j^{\underline{th}}$ column less the upper triangular set of zero elements have been mapped into v; therefore, it can intuitively be seen that for μ_{ij}, $1 \leq j \leq i \leq n$, and v_k, $1 \leq k \leq \hat{n}$, the mapping f can be defined by

$$k = i + n\,(j-1) - [j^2 - \tfrac{1}{2}\, j\,(j+1)] \tag{2.24}$$

which reduces to

$$k = i + (j-1)(2n-j)/2 \quad . \tag{2.25}$$

It must be shown that f, as defined by (2.24), is the mapping indicated by (2.23). Let $f(\mu_{ij}) = (v_k)$ be denoted by $f\,(i,j) = k$ for $1 \leq j \leq i \leq n$ and $1 \leq k \leq \hat{n}$. Therefore show that if $1 \leq j = \overline{j} \leq n - 1$ and $\overline{i} - i = 1$, or if $i = n$, $1 \leq j \leq n-1$, and $\overline{i} = \overline{j} = j+1$, then $\overline{k}-k = f(\overline{i},\overline{j})-f(i,j) = 1$.

Case I:

Let $1 \leq j = \overline{j} \leq n-1$ and $\overline{i}-i = 1$; this implies that

$$\begin{aligned} \overline{k}-k &= f(\overline{i},\overline{j}) - f(i,j) \\ &= \overline{i} + (\overline{j}-1)(2n-\overline{j})/2-i-(j-1)(2n-j)/2 \\ &= \overline{i} - i = 1 \quad . \end{aligned}$$

Case II:

Let $i=n$, $1 \leq j \leq n-1$, and $\overline{i}=\overline{j} = j+1$; this implies that

$$\begin{aligned} \overline{k}-k &= f(\overline{i}, \overline{j}) - f(i,j) \\ &= \overline{i} + (\overline{j}-1)(2n-\overline{j})/2 - i -(j-1)(2n-j)/2 \\ &= j+1 + (j+1-1)(2n-j-1)/2 \\ &\qquad -n - (j-1)(2n-j)/2 \\ &= j+1 + (2nj-j^2-j)/2 \\ &\quad -n - (2nj-j^2-2n+j)/2 \\ &= 1. \end{aligned}$$

Therefore f is the mapping indicated by (2.23). This mapping is defined in the subroutine INDEX which is used by BIGSET. It is obvious that f is one-to-one, and therefore f^{-1} exists. f^{-1} is defined by taking the elements of the solution vector v in order from left to right and placing them into the lower triangular matrix L in column order.

TTSTAR is then called to find the lower triangular matrix T_r such that $M_r = T_r T_r^*$.

Now that T_r, $\exp(\Delta tA)$, and T have been computed and the vectors $w^{(k)}$ for $k = 0,1,2, \cdots$ can be constructed from the w-sequence, z(0) can be computed by (1.26) and the sequence of vectors $z(k\Delta t)$ for $k = 1,2,\cdots$ can be computed by (1.29). The input vector SB is also known; therefore, the specified time series, x(t) for $t = 0, \Delta t, 2\Delta t, \cdots$, can be computed by (1.30). This process is carried out in the subroutine SOLUTION. This completes the construction of the specified time series.

The card input to GAUSSIAN is of two types, problem data and flags. The sole purpose of the flags is to control the flow of the program. The flag NOSKIP is read once and only once each time the program is compiled; all other input parameters must be read anew for each time series that is to be generated.

The following is the list of problem data:

N	n in equation (2.12). N is also used as a flag in that GAUSSIAN terminates if N = 0; this is to be used for normal exit.
M	m in equation (2.12).
SA(1),$\cdots$, SA(N+1)	$a_1, \cdots, a_{n+1}$ in equation (2.12).
SB(1),$\cdots$, SB(M+1)	$b_1, \cdots, b_{m+1}$ in equation (2.12).
DT	Δt, time interval at which samples of the time series are to be calculated.

The following is the list of flags:

NOSKIP	the number of end-of-files (ie., previously computed time series) which must be skipped on the output tape before the first time series is computed and then buffered out onto this output tape.
ISTOP	the number of points of the time series which are to be buffered out onto the output tape in each block (except for the last block which contains the remaining points). It is necessary that ISTOP $\leq$ 5604.
IPOWER T	used when factoring the matrix M.
IPOWER TR	used when factoring the matrix M_r. IPOWER T and IPOWER TR are called IPOWER in the subroutine TTSTAR. The check, "is $\lvert m_{ii} \rvert < 10^{-IPOWER}$?," is made before m_{ii} is used as a divisor in TTSTAR where m_{ii} are diagonal elements of M or M_r.
IPOWERB	used when solving the set of equations (1.23).
IPOWERC	used when solving the set of equations (1.28). IPOWERB and IPOWERC are called IPOWER in the subroutine GAUSS P. The check, "is $\lvert a_{ii} \rvert < 10^{-POWER}$?," is made before a_{ii} is used as a divisor in GAUSSP where a_{ii} are the diagonal elements of the constant matrix under consideration.
IFT	associated with IPOWERT.
IFTR	associated with IPOWER TR.
IFB	associated with IPOWERB.

IFC — associated with IPOWERC. Let IF denote any of the above four flags. If IF = 1 and a zero divisor has been detected by its IPOWER, a diagnostic will be given, but the calculation of this time series will continue. However, if IF = 0 and a zero divisor has been detected, a diagnostic will be given, but the program will go to the next time series.

IF N PRINT = 1 — print the number of points which are in the following block of time series points.

= 0, — do not print the number of points.

IF X PRINT = 1 — print the computed time series in blocks of ISTOP points.

= 0, — do not print the time series points.

The data deck to GAUSSIAN must be stacked as follows:

NO SKIP (Format I4)

<table>
<tr><td>N, M (Format 2I4)
SA(1),···, SA(N+1) (Format 4F20.6)
SB(1),···, SB(M+1) (Format 4F20.6)
DT (Format F20.6)
ISTOP, IPOWERT, IPOWERTR, IPOWERB, IPOWERC
(Format 5I4)
IFNPRINT, IFXPRINT, IFT, IFTR, IFB, IFC
(Format 6I1)</td><td>}</td><td>Repeated for each time series to be generated.</td></tr>
</table>

(Blank card; flags normal exit.)

The problem data are printed and labelled. The intermediate matrices and vectors, which are found by the various subroutines are printed

unlabelled as they are calculated.

More than one time series can be shaped from the same w-sequence each time GAUSSIAN is compiled. This input tape which contains the w-sequence must be of the same format as defined for the output tape from CLLCNVRT. Each block of the w-sequence should not exceed 588 points. GAUSSIAN terminates when and only when either N = 0 or a parity error has been detected on either the input or output tape.

The output tape has the following form:

```
ISTOP
X(1) · · · · X(ISTOP)
 .             .
 .             .
 .             .
ISTOP
X(1) · · · · X(ISTOP)
IPRINT
X(1) · · · · X(IPRINT)
    end-of-file

ISTOP
X(1) · · · · X(ISTOP)
 .             .
 .             .
 .             .
IPRINT
X(1) · · · · X(IPRINT)
    end-of-file.
```

X	array of time series points.
IPRINT	number time series points in the last block.

III. "WHITE SEQUENCE," w-SEQUENCE, AND EXAMPLE TIME SERIES

A spectral analysis program similar to the program SPECT, as found in a report by Ellis and Boston,[15] is used to find the autocorrelation function and power spectral density function of the "white sequence" generated by ISLASHO and four example time series generated by GAUSSIAN. The autocorrelation function is defined by (1.3) and the power spectral density function is defined by (1.4). In SPECT the time τ in (1.3) and (1.4) is taken at regular intervals $\Delta\tau$, and therefore

$$\tau = l\Delta\tau, \; l=0,1,2, \cdots, \tag{3.1}$$

where l is called the lag number. For practical applications on the computer, τ can only take on finite values and the sequences considered will only have a finite number of points; therefore a finite number of lags will be taken in SPECT. The maximum lag number will generally be about ten percent of the number of points in the given sequence.

A. "White Sequence"

The condition (1.14) states that the autocorrelation function of a "white sequence" must be zero for all but the zeroth lag. The "white sequence" under consideration is $\{e^n\}$, $1 \leq n \leq 4266$. It is also stated in Chapter I, Section B, that the power spectral density function of ν_n must be constant. The plots of the autocorrelation function and the power spectral density function of ν_n as computed by SPECT for time interval $\Delta\tau = .1$ sec. and $l = 200$ lags are shown in Figures 1 and 2, respectively. It can be seen from these two plots that the sequence $\{e^n\}$ is a good numerical approximation to a "white sequence."

B. w-Sequence

A short program, which is not listed, was written to find the mean and the variance of the w-sequence. Let the mean be $m = E(w_i)$ and the variance be defined as

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (w_i - m)^2 \quad , \tag{3.2}$$

where w_i, $1 \leq i \leq N$, are the N points of the w-sequence. It is stated in Chapter I that the w-sequence, as constructed by (1.17), should have mean 0 and variance 1. The numerical w-sequence constructed from $\{e^n\}$ by CLLCNVRT was found to have $m \approx -.029$ and $\sigma^2 \approx .9577$.

C. Four Example Time Series

Four example problems are now to be considered. The following three steps are taken in the consideration of these examples:

(1) Given the constant coefficients of the polynomials $P(i\omega)$ and $Q(i\omega)$ of the theoretical power spectral density function $V(\omega)$ as given by (2.12) and a time interval $\Delta\tau$, GAUSSIAN is used to shape a time series with this power spectral density function.

(2) A log-log plot of the theoretical power spectral density function as a function of frequency f, is made where $f = \frac{\omega}{2\pi}$ and ω is the angular velocity given in radians/sec.

(3) The actual autocorrelation function and power spectral density function are then calculated by SPECT; the autocorrelation function is plotted linearly against time while the power spectral density function is plotted against frequency on a log-log plot.

Only the power spectral density function is considered in the first three examples; however, both the autocorrelation and the power spectral

density functions for example four are given by Richie[16] and will therefore be considered.

In all cases the theoretical and actual log-log plots of the power spectral density functions were found to have approximately the same shape; however, for each example, the two curves were found to differ by some arbitrary constant. If the magnitude of the points of the power spectral density function is specified and not just the shape, the specified time series can be obtained by multiplying each point of the original time series by the square root of the constant whose logarithm is the difference between the theoretical and the actual curves.

It can be seen by (1.3) that if each point of the time series $x(t)$ is multipled by a scale factor s, then for every τ, the autocorrelation function is multiplied by s^2. Since the constant s^2 can be removed from the integral in (1.4), the power spectral density function is also multiplied by s^2.

Let t_i and a_i be corresponding points on the theoretical and actual power spectral density curves, respectively, and if there exists an $s > 0$ such that for every i where t_i and a_i are defined, $\log s^2 = \log t_i - \log a_i$, then $t_i = s^2 a_i$. The converse is also true. Therefore the specified magnitude of the power spectral density function can be obtained by multiplying each point of the time series by the scale factor s. This scale factor may have physical units.

In Figures 3 through 6 the crooked continuous curve is the actual power spectral density function as computed by SPECT and the smooth dotted curve is a multiple of the theoretical curve which gives an approximate fit to the points of the actual power spectral density function.

The theoretical power spectral density function of example one is given to be

$$V_1(\omega) = \frac{9\omega^2 + 1}{\omega^4 - 6\omega^2 + 25} \quad , \tag{3.3}$$

which has zeros at the points (0,1/3) and (0,-1/3) and poles at the points r, $\overline{r}$, - r, and -$\overline{r}$ where r = (2,1). Therefore for all real ω, the condition $0 < V_1(\omega) < \infty$ holds. Also for real ω, the conditions $V_1(\omega) = V_1(-\omega)$ and $V_1(\omega) \to \pm \infty$ hold; therefore the three conditions, as stated in (1.8), exist which insure that $V_1(\omega)$ can be represented in the form of (2.12). Specifically, $V_1(\omega)$ can be represented as

$$V_1(\omega) = \left| \frac{3(i\omega) + 1}{(i\omega)^2 + 2(i\omega) + 5} \right|^2 \tag{3.4}$$

where $\xi^2 + 2\xi + 5$ has zeros at (-1,2) and (-1,-2). Both of these zeros lie in the halfplane Re $\xi < 0$.

The theoretical power spectral density function of example two is given to be

$$V_2(\omega) = \frac{\omega^4 - 70\omega^2 + 1369}{\omega^6 + 14\omega^4 + 49\omega^2 + 36} \quad . \tag{3.5}$$

The numerator of $V_2(\omega)$ has zeros at the points r, $\overline{r}$, -r, and -$\overline{r}$ where r = (6,1), and the denominator of $V_2(\omega)$ is greater than zero for all real ω; therefore, for all real ω, the condition $0 < V_2(\omega) < \infty$ holds. Also the other two conditions of (1.8) hold; therefore $V_2(\omega)$ can be represented, as indicated in (2.12), by

$$V_2(\omega) = \left| \frac{(i\omega)^2 + 2(i\omega) + 37}{(i\omega)^3 + 6(i\omega)^2 + 11(i\omega) + 6} \right|^2 \tag{3.6}$$

where $\xi^3 + 6\xi^2 + 11\xi + 6$ has zeros at (-1,0), (-2,0), and (-3,0). All these zeros lie in the halfplane Re $\xi < 0$.

The theoretical power spectral density function of example three

is given to be

$$V_3(\omega) = \frac{400\omega^2 + 1}{\omega^6 + 14\omega^4 + 49\omega^2 + 36} \quad . \tag{3.7}$$

$V_3(\omega)$ has zeros at the points $(0, \frac{1}{20})$ and $(0, -\frac{1}{20})$. The denominator of this example is the same as in example two. It can be seen, as in examples one and two, that $V_3(\omega)$ can be represented by

$$V_3(\omega) = \left| \frac{20(i\omega) + 1}{(i\omega)^3 + 6(i\omega)^2 + 11(i\omega) + 6} \right|^2 . \tag{3.8}$$

Example four is an example problem which appears as channel 4 in a report by Richie.[17] The theoretical power spectral density function is given to be

$$V_4(\omega) = 4Ak\frac{\omega^2 + (k^2 + c^2)}{\omega^4 + 2(k^2-c^2)\omega^2 + (k^2 + c^2)^2} \tag{3.9}$$

where A = 2149.2, k = .658, and c = 2.71. Since A, c, and k are positive and k > c, the conditions (1.8) hold for real ω. Therefore, $V_4(\omega)$ can be represented as

$$V_4(\omega) = \left| \frac{b_1(i\omega) + b_2}{a_1(i\omega)^2 + a_2(i\omega) + a_3} \right|^2 \tag{3.10}$$

where $b_1 \approx 75.210999197$, $b_2 \approx 53.521755647$, $a_1 = 1.0$, $a_2 = 1.316$, and $a_3 = .506405$.

Both zeros of $a_1\xi^2 + a_2\xi + a_3$ lie in the halfplane $\mathrm{Re}\xi < 0$ since $a_1 > 0$, $a_2 > 0$, and $a_3 > 0$. This fact can be shown by letting $x = \mathrm{Re}\xi$ and $y = \mathrm{Im}\xi$ and then assuming that there exists a ξ with $x \geqq 0$ such that $a_1\xi^2 + a_2\xi + a_3 = 0$. This complex equation is equivalent to the following two real equations:

$$a_1(x^2-y^2) + a_2x+a_3 = 0 \tag{3.11}$$

and

$$2a_1xy + a_2y = 0 \quad . \tag{3.12}$$

By (3.12), $y = 0$ since $x \geq 0$. Then by (3.11), $a_1(x^2-y^2) + a_2x+a_3 \neq 0$ which is a contradiction; therefore both zeros of $a_1\xi^2 + a_2\xi + a_3$ lie in the halfplane $\mathrm{Re}\,\xi < 0$.

The actual power spectral density function of the time series generated by GAUSSIAN for this example was found to differ from the theoretical power spectral density function only by a multiplicative constant; however, the theoretical autocorrelation function given by Richie[18] and the actual autocorrelation function differed greatly. This illustrates the fact, as pointed out previously, that small perturbations in $V(\omega)$ may result in great changes of $R(\tau)$. These two autocorrelation functions are shown in Figure 7, and the power spectral density functions are shown in Figure 6.

The degree and coefficients of the polynomials P and Q, which are used as input parameters to GAUSSIAN, can be taken from equations (3.4), (3.6), (3.8), and (3.10). These parameters, along with the time intervals used in generating a time series for each of these examples, are also given in Table 1.

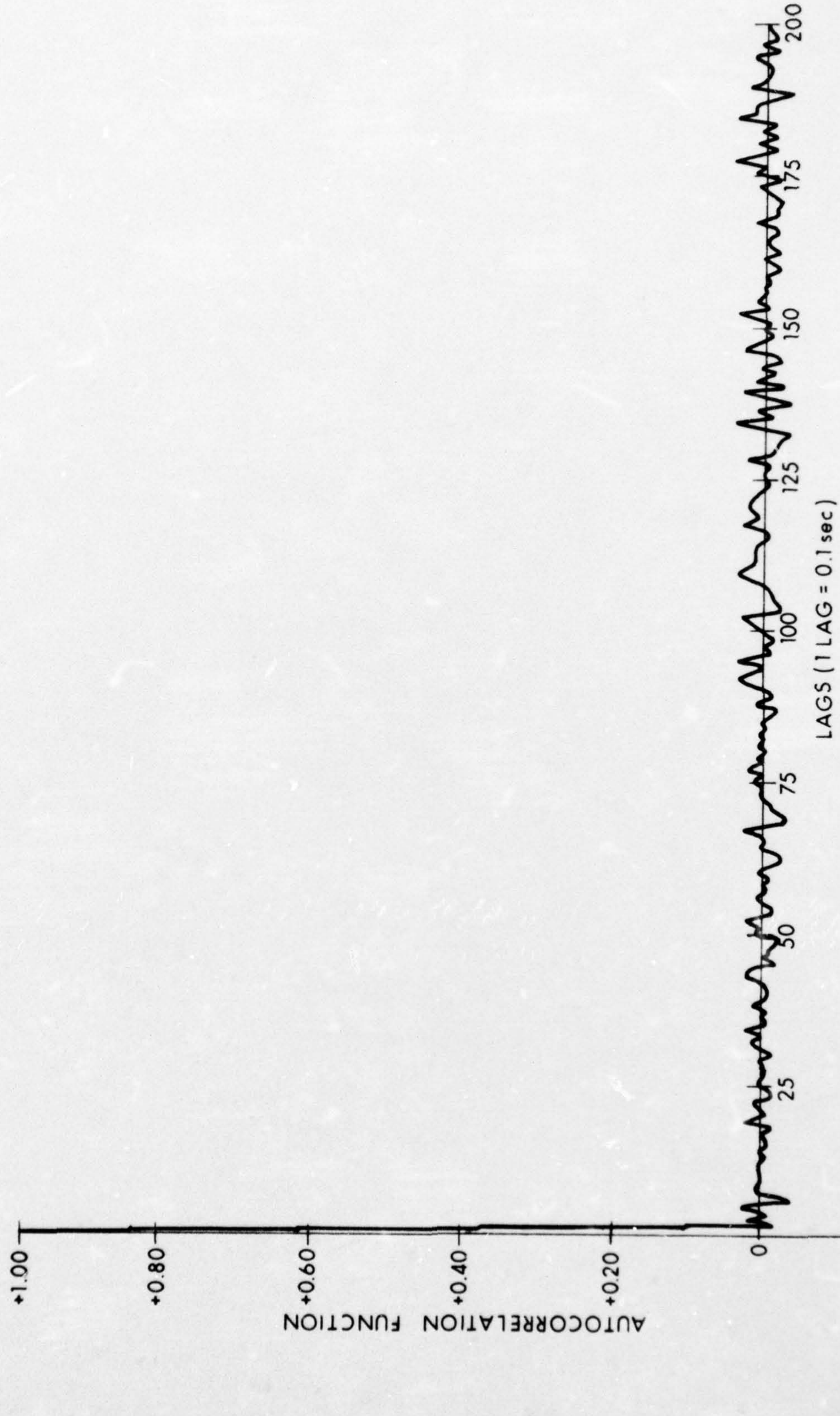


FIGURE 1
WHITE NOISE

DRL - UT
DWG AS - 66 - 688
WM - ORS
7 - 11 - 66

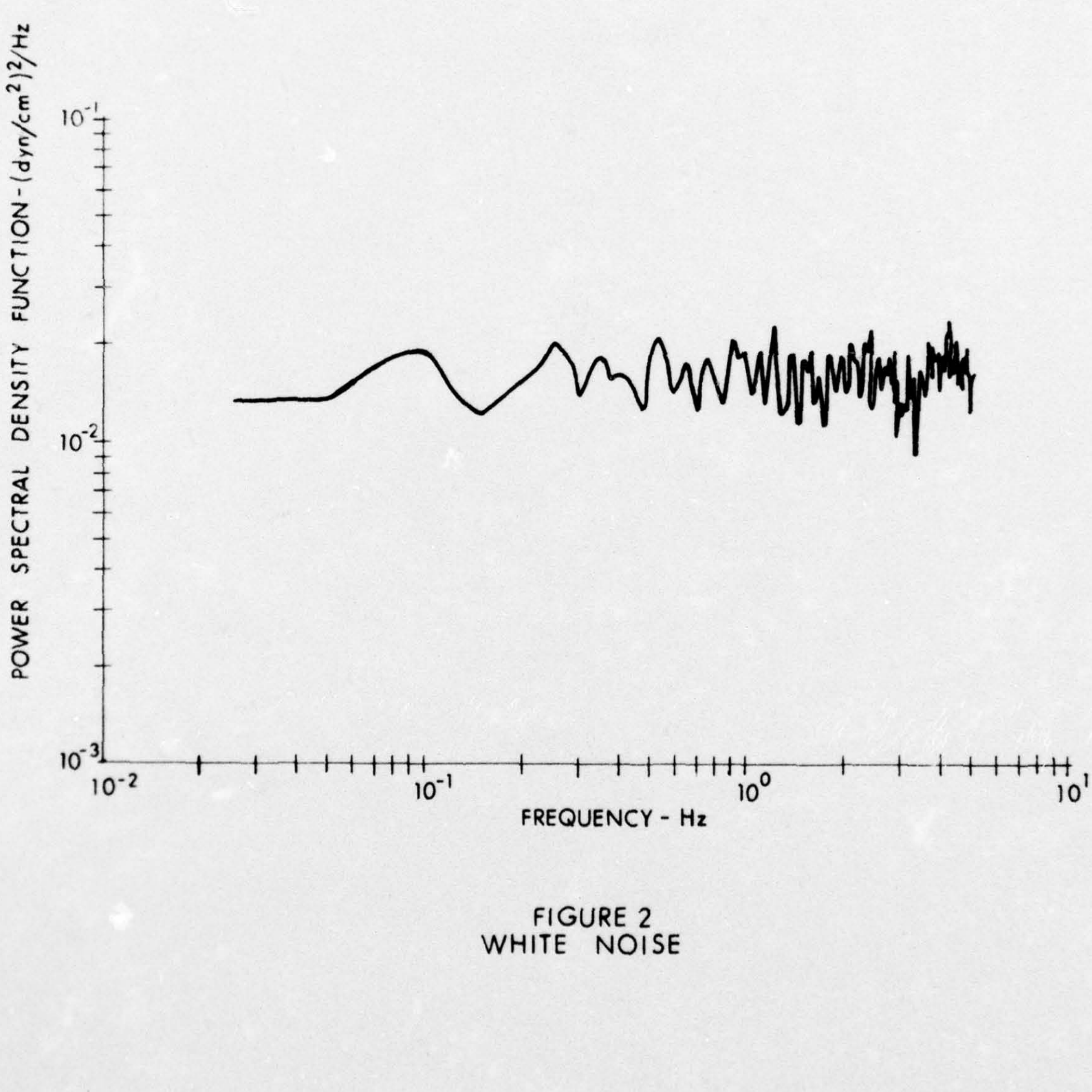


FIGURE 2
WHITE NOISE

DRL - UT
DWG AS-66-689
WM - ORS
7 - 11 - 66

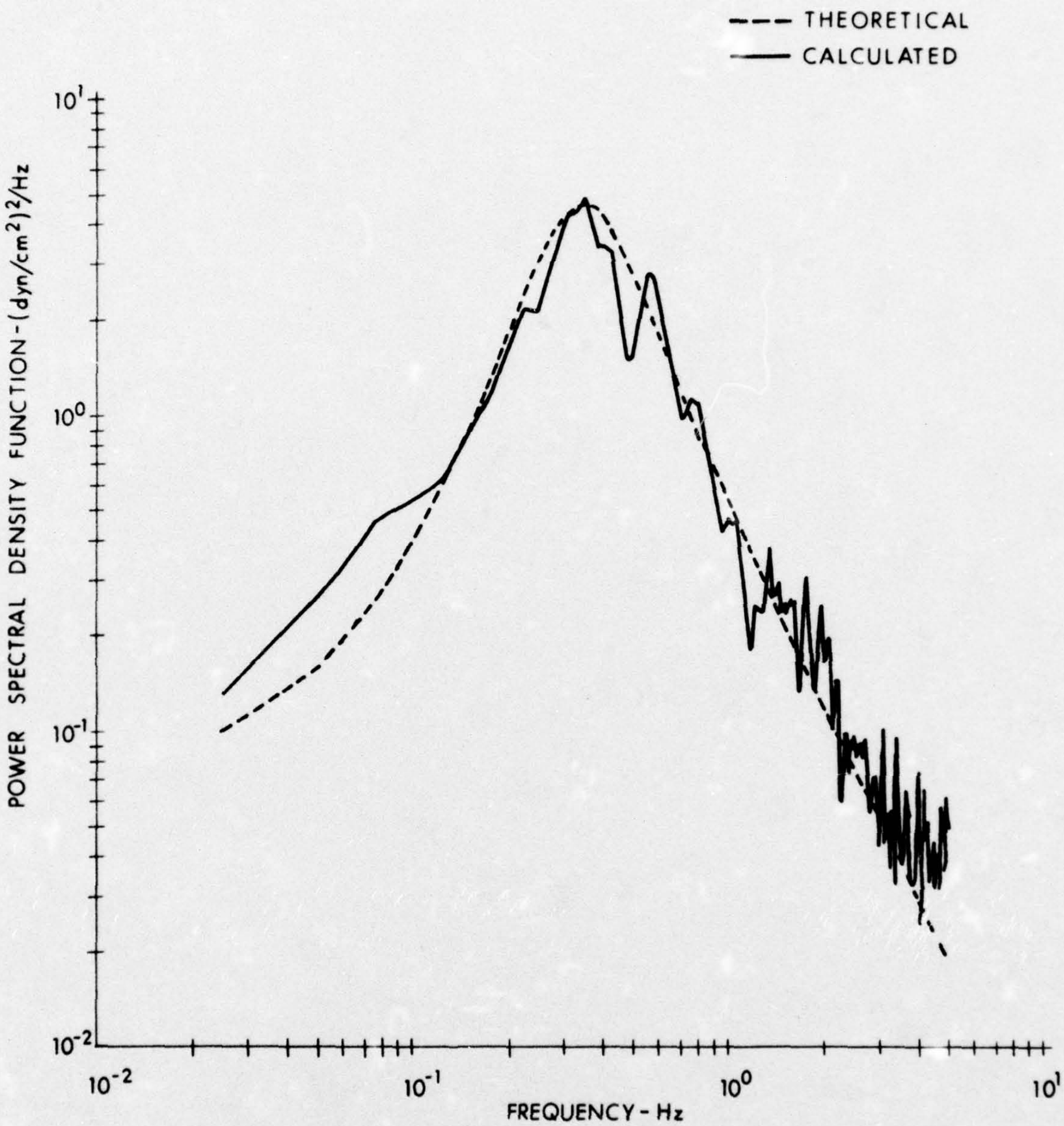


FIGURE 3
EXAMPLE 1

$$V_1(\omega) = \frac{9\omega^2 + 1}{\omega^4 - 6\omega^2 + 25}$$

DRL - UT
DWG AS - 66 - 690
WM - ORS
7 - 12 - 66

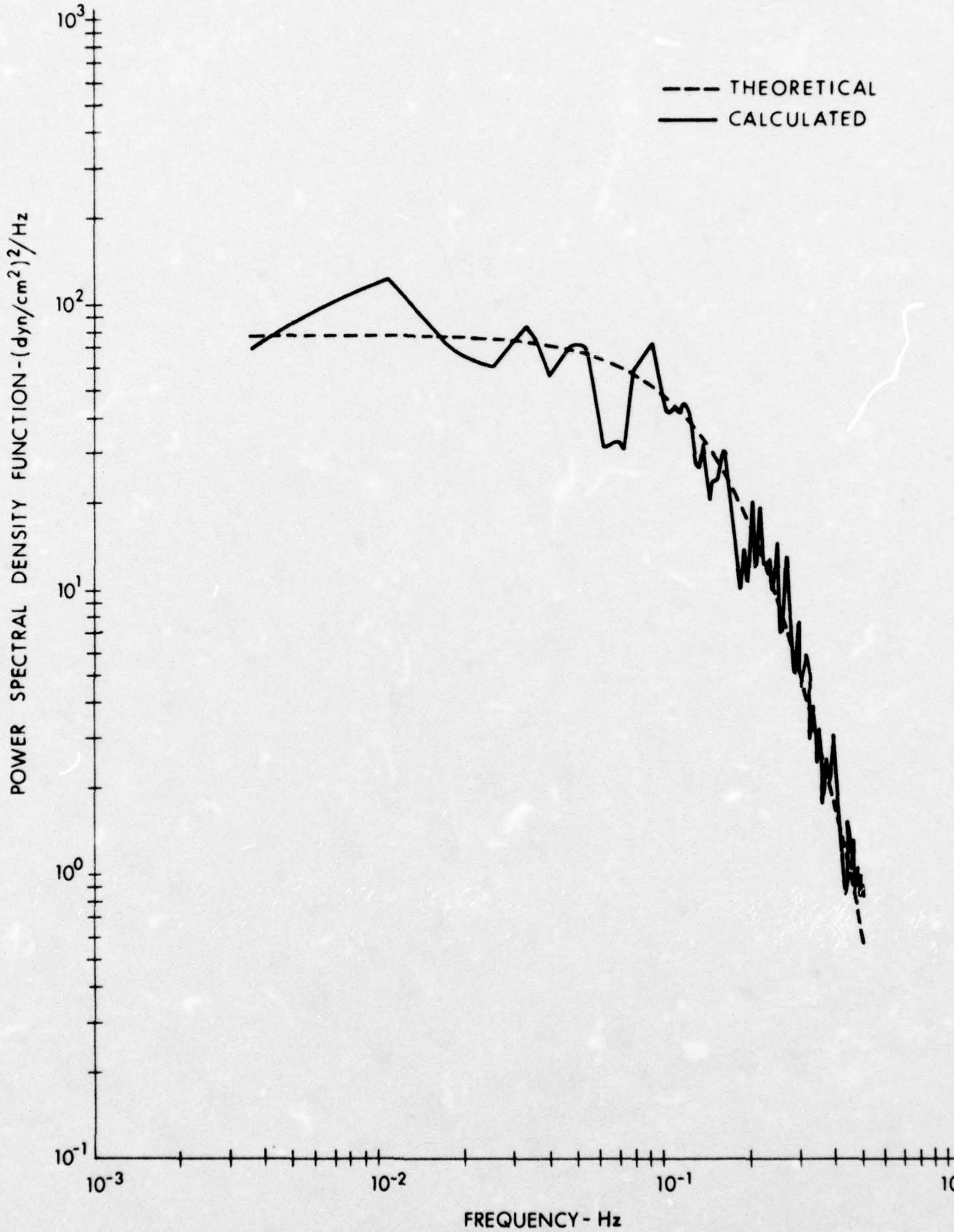


FIGURE 4
EXAMPLE 2

$$V_2(\omega) = \frac{\omega^4 - 70\omega^2 + 1369}{\omega^6 + 14\omega^4 + 49\omega^2 + 36}$$

DRL - UT
DWG AS-66-691
WM - ORS
7 - 12 - 66

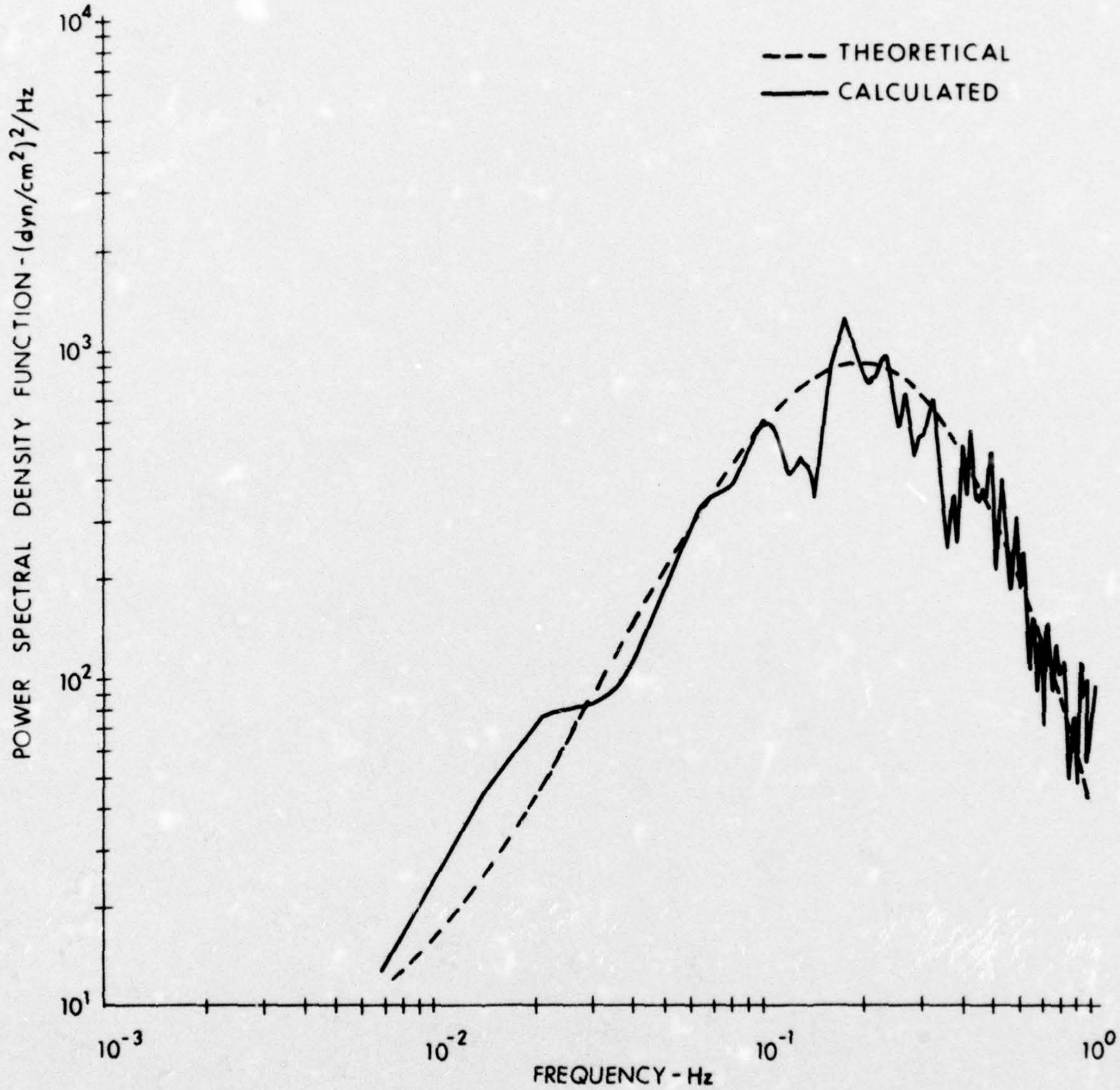


FIGURE 5
EXAMPLE 3

$$V_3(\omega) = \frac{400\omega^2 + 1}{\omega^6 + 14\omega \quad + 49\omega^2 + 36}$$

DRL - UT
DWG AS - 66 - 692
WM - ORS
7 - 13 - 66

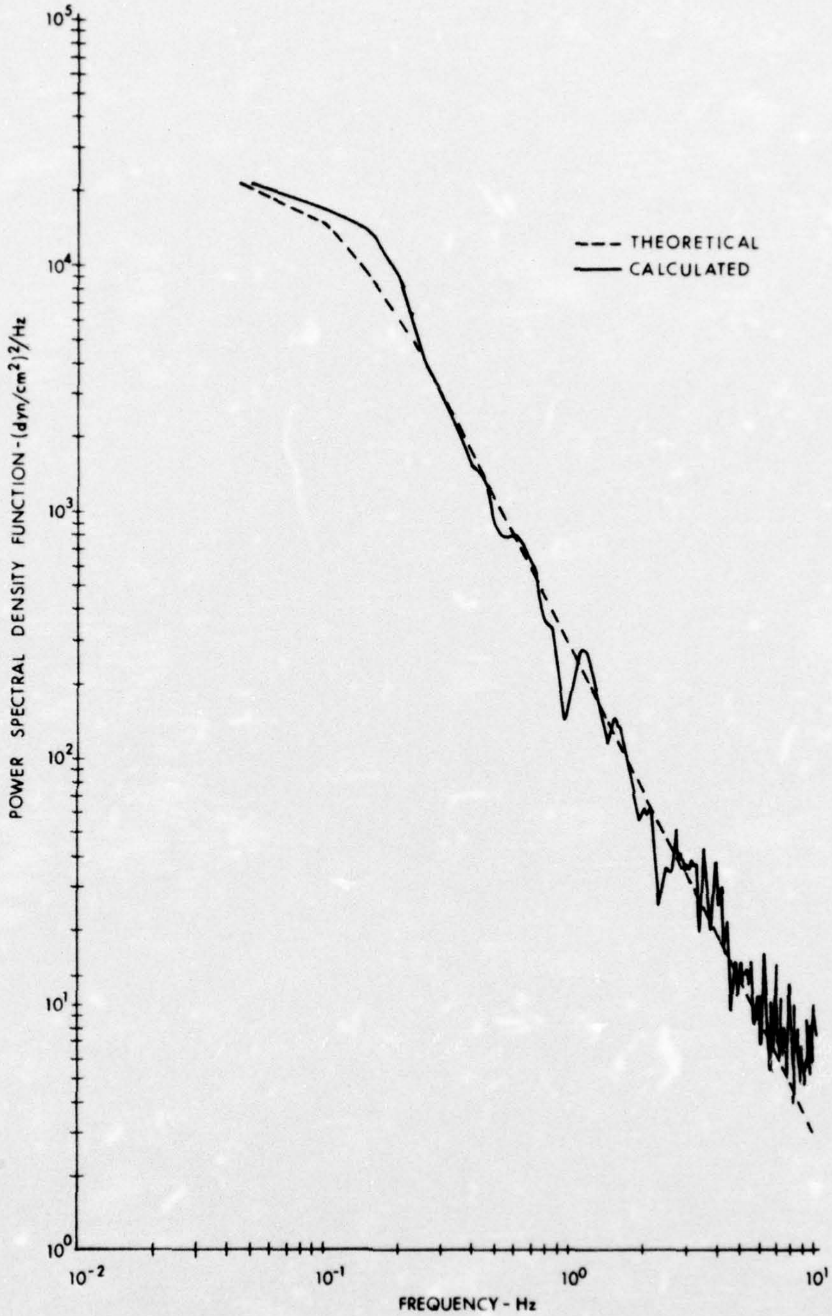


FIGURE 6
EXAMPLE 4
RICHIE'S SEQUENCE
CHANNEL 4

DRL - UT
DWG BS-66-693
WM - ORS
7 - 13 - 66

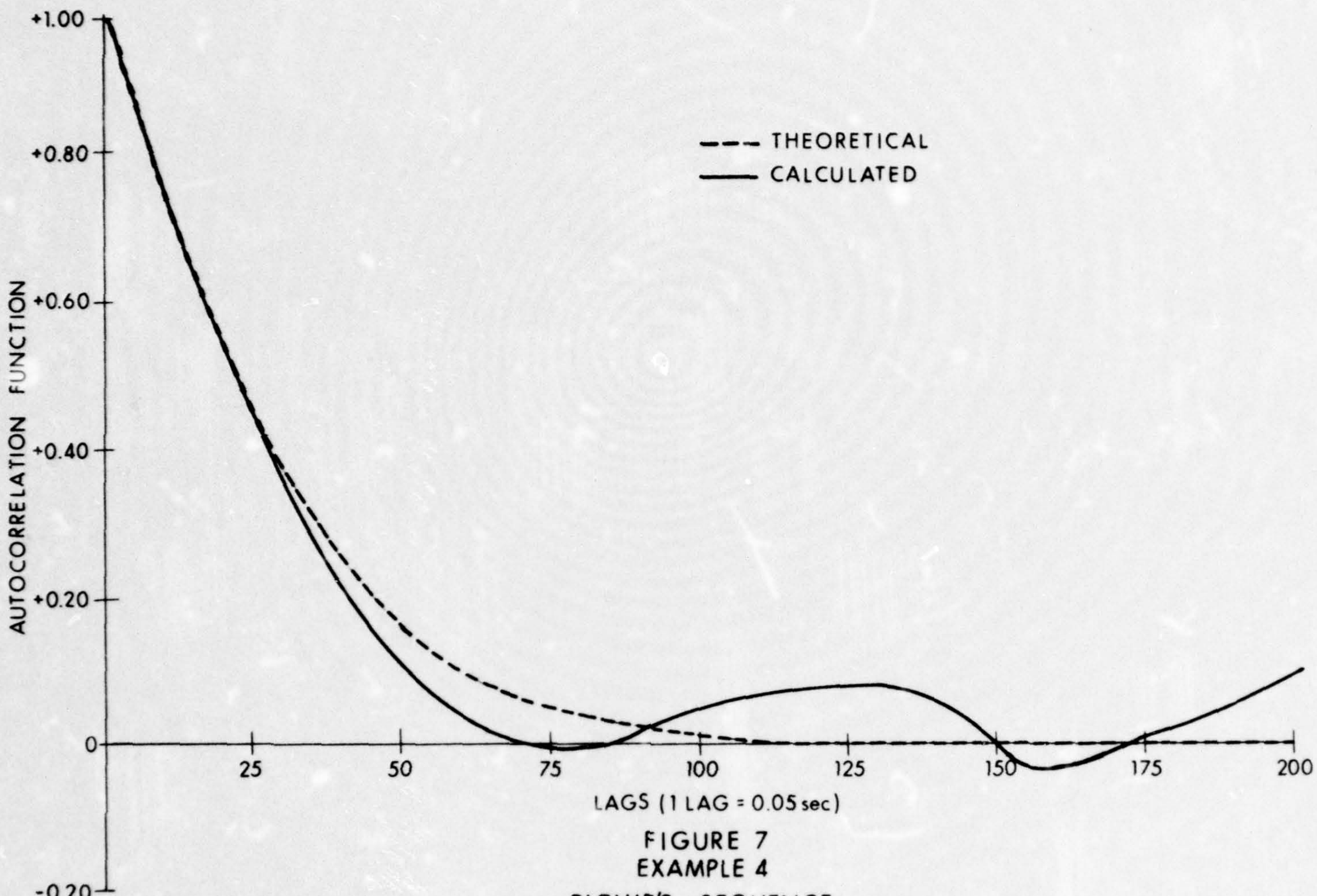


FIGURE 7
EXAMPLE 4
RICHIE'S SEQUENCE
CHANNEL 4

DRL - UT
DWG AS-66-694
WM - ORS
7 - 13 - 66

Table 1

	degree of P m	degree of Q n	coefficients of P	coefficients of Q	time interval $\Delta\tau$ (sec)	number of points in time series	number of lags in SPECT
Example 1	1	2	$b_1 = 3$ $b_2 = 1$	$a_1 = 1$ $a_2 = 2$ $a_3 = 5$	.1	2133	200
Example 2	2	3	$b_1 = 1$ $b_2 = 2$ $b_3 = 37$	$a_1 = 1$ $a_2 = 6$ $a_3 = 11$ $a_4 = 6$	1.0	1420	140
Example 3	1	3	$b_1 = 20$ $b_2 = 1$	$a_1 = 1$ $a_2 = 6$ $a_3 = 11$ $a_4 = 6$	.5	1420	140
Example 4	1	2	$b_1 = 75.210999197$ $b_2 = 53.521755647$	$a_1 = 1.0$ $a_2 = 1.316$ $a_3 = .506405$	.05	2133	200

```
      PROGRAM I SLASH 0
      DIMENSION X(500)
      COMMON/DATA/NO DEC, I DEC, IX1, IX2, IX3, L, ITHOUS, ICARRY
      COMMON IX(12000), IY(12000)
      ITHOUS = 1000
      REWIND 5
    1 FORMAT (16I5)
    2 FORMAT (//)
    3 FORMAT (1H1)
   98 FORMAT (2I1)
   16 FORMAT (6F20.11)
   17 FORMAT (2I10)
      TEN1=1.0E-1
      TEN2=1.0E-2
      TEN3=1.0E-3
      TEN4=1.0E-4
      TEN5=1.0E-5
      TEN6=1.0E-6
      TEN7=1.0E-7
      TEN8=1.0E-8
      TEN9=1.0E-9
      TEN10=1.0E-10
      TEN11=1.0E-11
      TEN12=1.0E-12
      PRINT 3
      READ 1, NO DEC
      READ 1, IX3, IX2, IX1
      READ 98, IF PRNT SQ, IF PRNT ME
      READ 17, ISTOP, IBUF
      IX(1)= IX1
      IX(2)= IX2
      IX(3)= IX3
      I DEC= NO DEC
      L = 3
      I COUNT =1
      I= I DEC/3
      IR = IDEC- I*3
      IF(IR) 5, 4
    4 X(1) = IX2*TEN3 + IX1*TEN6
      GO TO 6
    5 XX=IX3*10.0**(-IR)+IX2*10.0**(-3-IR)+IX1*10.0**(-6-IR)
      I=XX
      X(1)=XX-I
    6 I COUNT = I COUNT +1
    7 CALL RANDOM
      I= I DEC/3
      IR = I DEC -I*3
      IF(IR-1) 8,9,10
    8 X(ICOUNT)=IX(I)*TEN3+IX(I-1)*TEN6+IX(I-2)*TEN9+IX(I-3)*TEN12
      GO TO 12
    9 XX=IX(I+1)*TEN1+IX(I)*TEN4+IX(I-1)*TEN7+IX(I-2)*TEN10
      GO TO 11
   10 XX= IX(I+1)*TEN2+IX(I)*TEN5+IX(I-1)*TEN8+IX(I-2)*TEN11
   11 I= XX
      X(I COUNT) = XX-I
   12 IF(L .GT. ISTOP) 60, 14
   60 JJ = I COUNT/2
      JJ= JJ*2
      IF(I COUNT .EQ. JJ) 13, 6
   14 IF(I COUNT - IBUF) 6, 15
```

```
15 PRINT 1, I BUF
   IF (IF PRNT SQ) 30, 31
30 PRINT 16, (X(I), I=1, I BUF)
   PRINT 2
31 BUFFER OUT (5,1)(I BUF, I BUF)
32 GO TO (32, 33, 33, 77), UNITSTF(5)
33 BUFFER OUT(5,1)(X(1), X(I BUF))
34 GO TO (34, 35, 35, 77), UNITSTF(5)
35 I COUNT = 1
   GO TO 7
13 PRINT 1, I COUNT
   IF (IF PRNT SQ) 37, 38
37 PRINT 16, (X(I), I=1, I COUNT)
   PRINT 2
38 BUFFER OUT(5,1)(I COUNT, I COUNT)
39 GO TO (39, 40, 40, 77), UNITSTF(5)
40 BUFFER OUT(5,1)(X(1), X(I COUNT))
41 GO TO (41, 42, 42, 77), UNITSTF(5)
42 END FILE 5
   IY(1)=IX1
   IY(2)=IX2
   IY(3)=IX3
   PRINT 1, NO DEC
   PRINT 1, (IY(I), I=1, 3)
   PRINT 2
   PRINT 17, I DEC, L
   IF (IF PRNT ME) 43, 44
43 PRINT 1, (IX(I), I=1, L)
   PRINT 2
44 BUFFER OUT (5,1)(I DEC, I DEC)
45 GO TO (45, 46, 46, 77), UNITSTF(5)
46 BUFFER OUT( 5,1)(L,L)
47 GO TO (47, 48, 48, 77), UNITSTF(5)
48 BUFFER OUT(5,1)(IX(1), IX(L))
49 GO TO (49, 50, 50, 77), UNITSTF(5)
50 END FILE 5
   GO TO 99
77 PRINT 78
78 FORMAT (25H PARITY ERROR ON MAG TAPE)
99 CONTINUE
      END

      3200 FORTRAN DIAGNOSTIC RESULTS - FOR    ISLASHO

NO ERRORS
COMPASS,L,X
```

```
COMPASS-32      (2.1)                                 RANDOM
                                                      ENTRY      RANDOM   02/03/66  PAGE     2
      00000   01077777 01 0  77777 0  RANDOM    UJP        **
      00001   14600000 14 1  00000 2            ENA        0
      00002   40000007 40 0 D00007 0            STA        CARRY
      00003   14177776 14 0  77776 1            ENI        -1,1
      00004   20000005 20 0 D00005 0            LDA        L
      00005   15477776 15 1  77776 0            INA,S      -1
      00006   44000007 44 0 P00007 0            SWA        *+1
      00007   10177777 10 0  77777 1  INDEX1    IST        **,1
      00010   01000012 01 0 P00012 0            UJP        *+2
      00011   01000022 01 0 P00022 0            UJP        FINISH1
      00012   20100000 20 0 C00000 1            LDA        IX,1
      00013   50000002 50 0 D00002 0            MUA        IX1
      00014   30000007 30 0 D00007 0            ADA        CARRY
      00015   13077747 13 0  77747 0            SHAQ       -24
      00016   51000006 51 0 D00006 0            DVA        THOUSAND
      00017   41127340 41 0 C27340 1            STQ        IY,1
      00020   40000007 40 0 D00007 0            STA        CARRY
      00021   01000007 01 0 P00007 0            UJP        INDEX1
      00022   20000005 20 0 D00005 0  FINISH1   LDA        L
      00023   53500000 53 1  00000 1            IAI        1
      00024   20000007 20 0 D00007 0            LDA        CARRY
      00025   40127340 40 0 C27340 1            STA        IY,1
      00026   14600000 14 1  00000 2            ENA        0
      00027   40000007 40 0 D00007 0            STA        CARRY
      00030   14177776 14 0  77776 1            ENI        -1,1
      00031   20000005 20 0 D00005 0            LDA        L
      00032   15477776 15 1  77776 0            INA,S      -1
      00033   44000034 44 0 P00034 0            SWA        *+1
      00034   10177777 10 0  77777 1  INDEX2    IST        **,1
      00035   01000037 01 0 P00037 0            UJP        *+2
      00036   01000050 01 0 P00050 0            UJP        FINISH2
      00037   20100000 20 0 C00000 1            LDA        IX,1
      00040   50000003 50 0 D00003 0            MUA        IX2
      00041   30000007 30 0 D00007 0            ADA        CARRY
      00042   30127341 30 0 C27341 1            ADA        IY+1,1
      00043   13077747 13 0  77747 0            SHAQ       -24
      00044   51000006 51 0 D00006 0            DVA        THOUSAND
      00045   41127341 41 0 C27341 1            STQ        IY+1,1
      00046   40000007 40 0 D00007 0            STA        CARRY
      00047   01000034 01 0 P00034 0            UJP        INDEX2
      00050   20000005 20 0 D00005 0  FINISH2   LDA        L
      00051   15600001 15 1  00001 2            INA        1
      00052   53500000 53 1  00000 1            TAI        1
      00053   20000007 20 0 D00007 0            LDA        CARRY
      00054   40127340 40 0 C27340 1            STA        IY,1
      00055   14600000 14 1  00000 2            ENA        0
      00056   40000007 40 0 D00007 0            STA        CARRY
      00057   14177776 14 0  77776 1            ENI        -1,1
      00060   20000005 20 0 D00005 0            LDA        L
      00061   15477776 15 1  77776 0            INA,S      -1
      00062   44000063 44 0 P00063 0            SWA        *+1
      00063   10177777 10 0  77777 1  INDEX3    IST        **,1
      00064   01000066 01 0 P00066 0            UJP        *+2
```

```
COMPASS-32      (2.1)                    RANDOM
        00065  01000077 01 0 P00077 0            UJP        FINISH3   02/03/66  PAGE     3
        00066  20100000 20 0 C00000 1            LDA        IX,1
        00067  50000004 50 0 D00004 0            MUA        IX3
        00070  30000007 30 0 D00007 0            ADA        CARRY
        00071  30127342 30 0 C27342 1            ADA        IY+2,1
        00072  13077747 13 0  77747 0            SHAQ       -24
        00073  51000006 51 0 D00006 0            DVA        THOUSAND
        00074  41127342 41 0 C27342 1            STQ        IY+2,1
        00075  40000007 40 0 D00007 0            STA        CARRY
        00076  01000063 01 0 P00063 0            UJP        INDEX3
        00077  20000007 20 0 D00007 0  FINISH3   LDA        CARRY
        00100  03100105 03 0 P00105 1            AZJ,NE     ZERO
        00101  20000005 20 0 D00005 0            LDA        L
        00102  15600002 15 1  00002 2            INA        2
        00103  40000005 40 0 D00005 0            STA        L
        00104  01000114 01 0 P00114 0            UJP        CHANGE
        00105  20000005 20 0 D00005 0  ZERO      LDA        L
        00106  15600002 15 1  00002 2            INA        2
        00107  53500000 53 1  00000 1            TAI        1
        00110  15600001 15 1  00001 2            INA        1
        00111  40000005 40 0 D00005 0            STA        L
        00112  20000007 20 0 D00007 0            LDA        CARRY
        00113  40127340 40 0 C27340 1            STA        IY,1
        00114  14177776 14 0  77776 1  CHANGE    ENI        -1,1
        00115  20000005 20 0 D00005 0            LDA        L
        00116  15477776 15 1  77776 0            INA,S      -1
        00117  44000120 44 0 P00120 0            SWA        *+1
        00120  10177777 10 0  77777 1  INDEX4    IST        **,1
        00121  01000123 01 0 P00123 0            UJP        *+2
        00122  01000126 01 0 P00126 0            UJP        FINISH4
        00123  20127340 20 0 C27340 1            LDA        IY,1
        00124  40100000 40 0 C00000 1            STA        IX,1
        00125  01000120 01 0 P00120 0            UJP        INDEX4
        00126  20000001 20 0 D00001 0  FINISH4   LDA        IDEC
        00127  30000000 30 0 D00000 0            ADA        NODEC
        00130  40000001 40 0 D00001 0            STA        IDEC
        00131  01000000 01 0 P00000 0            UJP        RANDOM
                                                 DATA
        00000                          NODEC     BSS        1
        00001                          IDEC      BSS        1
        00002                          IX1       BSS        1
        00003                          IX2       BSS        1
        00004                          IX3       BSS        1
        00005                          L         BSS        1
        00006                          THOUSAND  BSS        1
        00007                          CARRY     BSS        1
                                                 COMMON
        00000                          IX        BSS        12000
        27340                          IY        BSS        12000
                                                 END

                                                NUMBER OF LINES WITH DIAGNOSTICS
```

3200 FORTRAN (2.1) 03/03/66

```
      PROGRAM CLL CNVRT
    1 FORMAT (I1)
    2 FORMAT (28H PARITY ERROR ON OUTPUT TAPE)
    3 FORMAT (27H PARITY ERROR ON INPUT TAPE)
    4 FORMAT (//////)
      READ 1, IF PRINT
      CALL CONVERT (IF PRINT, I PARITY I,  I PARITY O)
      IF(I PARITY I .EQ. 1) 10, 11
   10 PRINT 3
      PRINT 4
      GO TO 99
   11 IF (I PARITY O .EQ. 1) 12, 99
   12 PRINT 2
      PRINT 4
   99 CONTINUE
         END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR CLLCNVRT

NO ERRORS

3200 FORTRAN (2.1) 03/03/66

```
      SUBROUTINE CONVERT( IF PRINT, I PARITY I, I PARITY O)
      DIMENSION X(500)
      REAL LOGF
    1 FORMAT (I10)
    2 FORMAT (6E20.10)
    3 FORMAT (//)
    5 FORMAT (1H1)
      REWIND 1
      REWIND 5
      I PARITY I = 0
      I PARITY O = 0
      PRINT 5
   20 BUFFER IN (5,1)(I BUF, IBUF)
   21 GO TO (21, 22, 23, 77), UNITSTF(5)
   23 END FILE 1
      PRINT 5
      RETURN
   77 I PARITY I = 1
      RETURN
   22 I BUF = I BUF
      BUFFER IN (5,1)(X(1),X(I BUF))
   24 GO TO (24, 25, 23, 77), UNITSTF(5)
   25 I BUF 2 = I BUF/2
      PI = 3.1415926536
      TPI = 2.0* PI
      DO 28 I=1, I BUF 2
      I2=I*2
      I2M1= I2-1
      X1= SQRTF(-2.0*LOGF(X(I2M1)))
      X2= TPI*X(I2)
      X(I2M1)=X1*COSF(X2)
   28 X(I2)=X1*SINF(X2)
      BUFFER OUT (1,1)( I BUF, I BUF)
   31 GO TO (31, 32, 32, 78), UNITSTF(1)
   78 I PARITY O = 1
      RETURN
   32 I BUF = I BUF
      BUFFER OUT (1,1)(X(1), X(I BUF))
   33 GO TO (33, 34, 34, 78), UNITSTF(1)
   34 IF( IF PRINT .EQ. 1) 40, 20
   40 PRINT 1, I BUF
      PRINT 2, (X(I), I=1, I BUF)
      PRINT 3
      GO TO 20
         END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR CONVERT

```
NO ERRORS
EQUIP,1=MT
LOAD,56
RUN
```

```
      PROGRAM GAUSSIAN
      COMMON SA(13), SB(12), C(78, 79), X(78), T(12, 12), F(12, 12),
     1         A(12, 12), B(12, 12)
    1 FORMAT (5I4)
    2 FORMAT (4F20.6)
    3 FORMAT (///, 25H THE B MATRIX IS SINGULAR)
    4 FORMAT (///)
    5 FORMAT (6F20.10)
    6 FORMAT (///, 26H BIGSET MATRIX IS SINGULAR)
    7 FORMAT (///, 27H PARITY ERROR ON INPUT TAPE)
    8 FORMAT (///, 28H PARITY ERROR ON OUTPUT TAPE)
    9 FORMAT(///, 23H T TSTAR DOES NOT EXIST)
   10 FORMAT(///, 25H TR TRSTAR DOES NOT EXIST)
   11 FORMAT (6I1)
   12 FORMAT (14H DETERMINANT = ,E20.10,//)
  100 FORMAT (1H1)
   66 FORMAT(36H CONSTANTS OF DENOMINATOR POLYNOMIAL  )
   67 FORMAT (34H CONSTANTS OF NUMERATOR POLYNOMIAL  )
   68 FORMAT (8H    N    M  )
   69 FORMAT (5X, 13HTIME INTERVAL  )
   70 FORMAT (10X, 5H(SEC)   )
      REWIND 2
      READ 1, NO SKIP
      DO 61 I=1, NO SKIP
   63 BUFFER IN (2,1)(ISKIP, ISKIP)
   62 GO TO(62, 63, 61, 64), UNITSTF(2)
   64 PRINT 65
   65 FORMAT(40H PARITY ERROR ON MAG TAPE WHILE SKIPPING  )
      GO TO 63
   61 CONTINUE
   60 READ 1, N, M
      IF (N .EQ. 0) 99, 71
   71 NP1 = N+1
      MP1= M+1
      PRINT 100
      PRINT 68
      PRINT 1, N, M
      READ 2, (SA(I), I=1, NP1)
      READ 2, (SB(I), I=1, MP1)
      READ 2, DT
      READ 1, ISTOP, IPOWERT, IPOWERTR, IPOWERB, IPOWERC
      READ 11, IF N PRINT, IF X PRINT, IFT, IFTR, IFB, IFC
      PRINT 4
      PRINT 66
      PRINT 5, (SA(I), I=1, NP1)
      PRINT 4
      PRINT 67
      PRINT 5, (SB(I), I=1, MP1)
      PRINT 4
      PRINT 69
      PRINT 70
      PRINT 5, DT
      PRINT 4
      CALL FIND B(N)
      DO 50 I=1, N
   50 PRINT 5, (C(I,J), J=1, NP1)
      PRINT 4
      CALL GAUSS P (N, KK, IPOWERB, DET)
      PRINT 12, DET
```

```
      IF(KK) 21, 20
   21 PRINT 3
      IF(IFB .EQ. 1) 20, 60
   20 CALL FIND CM(N)
      PRINT 5, (X(I), I=1, N)
      PRINT 4
      DO 51 I=1, N
   51 PRINT 5, (C(I,J), J=1, N)
      CALL T T STAR ( N, KK, IPOWERT)
      IF(KK) 41, 40
   41 PRINT 9
      IF (IFT .EQ. 1) 40, 60
   40 CALL MAKE A(N)
      PRINT 4
      DO 52 I=1, N
   52 PRINT 5, (T(I,J), J=1, N)
      PRINT 4
      DO 53 I=1, N
   53 PRINT 5, (A(I,J), J=1, N)
      CALL EXP A DT (N, DT)
      PRINT 4
      DO 54 I=1, N
   54 PRINT 5, (E(I,J), J=1, N)
      CALL RIGSET(N)
      NN= (N*(N+1))/2
      NNP1= NN+1
      PRINT 4
      DO 55 I=1, NN
   55 PRINT 5, (C(I,J), J=1, NNP1)
      CALL GAUSS P (NN, KK, IPOWERC, DET )
      PRINT 4
      PRINT 12, DET
      IF(KK) 22, 23
   22 PRINT 6
      IF (IFC .EQ. 1) 23, 60
   23 L=0
      PRINT 4
      PRINT 5, (X(I), I=1, NN)
      DO 24 I=1, N
      DO 24 J=I, N
      L=L+1
      C(I,J)= X(L)
   24 C(J,I)= X(L)
      PRINT 4
      DO 56 I=1, N
   56 PRINT 5, (C(I,J), J=1, N)
      DO 25 I=1, N
      DO 25 J=1, N
   25 A(I,J)= T(I,J)
      CALL T T STAR(N, KK, IPOWERTR)
      IF(KK) 43, 42
   43 PRINT 10
      IF (IFTR .EQ. 1) 42, 60
   42 REWIND 1
      PRINT 4
      DO 57 I=1, N
   57 PRINT 5, (T(I, J), J=1, N)
      PRINT 100
      CALL SOLUTION(N,M,IPARITYI,IPARITYO,ISTOP,IFNPRINT,IFXPRINT)
      IF(IPARITYI) 26, 98
   26 PRINT 7
      GO TO 99
```

```
98 IF( IPARITY0) 27, 60
27 PRINT 8
99 PRINT 100
   END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR   GAUSSIAN

UNDEFINED SIMPLE VARIABLES
    ISKIP

```
      SUBROUTINE FINDB(N)
      COMMON SA(13),SB(12),B(78,79)
      INTEGER Q, QM1
      NM1= N-1  $  NP1= N+1
      DO 1 I=1, NM1
    1 B(I, NP1)= 0.0
      B(N, NP1)= 0.5
      DO 2 I=1, N
      DO 2 J=1, N
    2 B(I, J)= 0.0
      DO 3 K=1, N
      KM1= K-1  $  KL=KM1/2.0 +.51 +1
      KU= (N+KM1)/2+1  $  A3= (-1.0)**KM1
      DO 3 Q=KL, KU
      QM1= Q-1  $  K4= N-2*QM1+K
    3 B(K,Q)=A3*(-1.0)**QM1*SA(K4)
        END

      3200 FORTRAN DIAGNOSTIC RESULTS - FOR   FINDB

NO ERRORS
```

3200 FORTRAN (2.1) 31/03/66

```
      SUBROUTINE FINDCM(N)
      COMMON SA(13),SB(12),CM(78,79),SM(78)
      DO 1 I=1, N
      DO 1 J=1, N
    1 CM(I,J)= 0.0
      DO 2 I=1, N
      A= -1.0
      DO 2 J=I, N, 2
      A= -A  $  K= (I+J)/2
      CM(I,J)= A*SM(K)
    2 CM(J,I)= CM(I,J)
        END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR FINDCM

NO ERRORS

```
      SUBROUTINE T T STAR (N, KK, IPOWERT)
      COMMON SA(13),SB(12),CM(78,79),SM(78),T(12,12)
C*********************************************************************************
C            THIS SUBROUTINE IS DESIGNED TO FACTOR A POSITIVE DEFINITE,
C     REAL, SYMMETRIC MATRIX, CM, INTO A LOWER-TRIANGULAR MATRIX, T,
C     WITH POSITIVE ELEMENTS ON THE MAIN DIAGONAL, SUCH THAT CM=T T*.
C*********************************************************************************
C           KK=0 IF THE TRIANGULAR MATRIX SOLUTINN EXISTS, IE. ALL THE
C     ELEMENTS ON THE MAIN DIAGONAL OF THE CM MATRIX ARE POSITIVE.
C           KK=1 IF THE SOLUTION DOES NOT EXIST.
C*********************************************************************************
      KK=0
      TEN POWER = 10.0**(-IPOWERT)
      DO 6 I=1, N
      IF (CM(I,I) .LT. TEN POWER) 7, 6
    7 KK=1
      GO TO 8
    6 CONTINUE
    8 NM1 = N-1
      DO 5 I=1, NM1
      IP1= I+1
      DO 5 J=IP1, N
    5 T(I,J)= 0.0
      T(1,1)= SQRTF(CM(1,1))
      DO 1 I=2, N
    1 T(I,1)= CM(I,1)/T
      DO 2 J=2, N
      SUM= 0.0  $  JM1= J-1
      DO 3 K=1, JM1
    3 SUM= SUM+ T(J,K)*T(J,K)
      T(J,J)=SQRTF(CM(J,J)-SUM)
      JP1= J+1
      DO 2 I=JP1, N
      SUM=0.0
      DO 4 K=1, JM1
    4 SUM= SUM+T(I,K)*T(J,K)
    2 T(I,J)=(CM(I,J)-SUM)/T(J,J)
        END

         3200 FORTRAN DIAGNOSTIC RESULTS - FOR    TTSTAR

NO ERRORS
```

3200 FORTRAN (2.1) 31/03/66

```
      SUBROUTINE MAKE A(N)
      COMMON SA(13),SB(12),C(78,79),X(78),T(12,12),F(12,12),A(12,12)
      NM1= N-1
      NP2= N+2
      DO 1 I=1, NM1
      DO 1 J=1, N
    1 A(I,J)= 0.0
      DO 3 I=1, NM1
      IP1= I+1
    3 A(I,IP1)=1.0
      DO 4 I=1, N
      NI= NP2-I
    4 A(N,I)= -SA(NI)
         END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR MAKEA

NO ERRORS

```
      SUBROUTINE EXP A DT(N,DT)
      COMMON SA(13),SH(12),AN(12,12),B(12,12),DUMMY(5874),X(78),T(12,12)
     1       ,E(12,12), A(12,12)
      NM1 = N-1
      FACN= 1.0
      DO 1 I=1, N
      DO 1 J=1, N
      AN(I,J)= A(I,J)
    1 E(I,J)= 0.0
      DO 2 I=1, N
    2 E(I,I)= 1.0
      EPSCHK= 0.0
      DO 7 I=1, N
      CHECK= ABSF(A(N,I))
      IF(CHECK .GT. EPSCHK) 8,7
    8 EPSCHK= CHECK
    7 CONTINUE
      EPSCHK = EPSCHK*1.0E-12
    5 TFN= DT/FACN
      DO 12 I=1, N
      DO 12 J=1, N
   12 B(I,J)= AN(I,J)*TFN
      DO 28 J=1, N
      DO 28 I=1, N
      IF(ABSF(B(I,J)) .LT. EPSCHK) 28, 9
   28 CONTINUE
      GO TO 99
    9 DO 6 I=1, N
      DO 6 J=1, N
    6 E(I,J)= E(I,J)+B(I,J)
      DO 11 I=1, NM1
      IP1 = I+1
      DO 11 J=1, N
   11 AN(I,J) = B(IP1,J)
      BNN = B(N,N)
      AN(N,1) = BNN*A(N,1)
      DO 10 I=1, NM1
      IP1 = I + 1
   10 AN(N,IP1) = B(N,I)+BNN*A(N,IP1)
      FACN= FACN+1.0
      GO TO 5
   99 CONTINUE
         END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR EXPADT

NO ERRORS
LOAD,56
RUN

```
      SUBROUTINE GAUSS P (N, IF SINGLR, IPOWER, DET)
      COMMON SA(13), SB(12), A(78, 79), X(78)
      REAL M
      INTEGER R, RP1, RMAX
      ISIGN = 1
      NM1 = N-1
      NP1 = N+1
      IF SINGLR = 0
      TEN POWER = 10.0**(-IPOWER)
      DO 1 R=1, NM1
      AMAX = ABSF(A(R,R))
      RMAX = R
      RP1 = R+1
      DO 3 I=RP1, N
      ABSFA = ABSF(A(I,R))
      IF( ABSFA .GT. AMAX) 2, 3
    2 AMAX = ABSFA
      RMAX = I
    3 CONTINUE
      IF(AMAX .LT. TEN POWER) 5, 4
    5 IF SINGLR = 1
    4 IF (R .EQ. RMAX) 10, 9
    9 DO 6 I=R, N
      ATEMP = A(R,I)
      A(R,I)= A(RMAX,I)
    6 A(RMAX,I) = ATEMP
      BTEMP = A(R,NP1)
      A(R,NP1) = A(RMAX, NP1)
      A(RMAX,NP1) = BTEMP
      ISIGN = -ISIGN
   10 DO 1 I=RP1, N
      M = -A(I,R)/A(R,R)
      DO 7 J=RP1, N
    7 A(I,J) = A(I,J)+ M*A(R,J)
    1 A(I,NP1) = A(I, NP1) + M*A(R, NP1)
      IF (ABSF(A(N,N)) .LT. TENPOWER) 11, 8
   11 IF SINGLR = 1
    8 CONTINUE
      DO 21 I=1, N
      IM1 = I - 1
      NN = N-IM1
      XX = A(NN, NP1)
      DO 22 J=1, IM1
      NP1J = NP1 - J
   22 XX = XX-A(NN,NP1J)*X(NP1J)
   21 X(NN) = XX/A(NN,NN)
      DET = 1.0
      DO 31 I=1, N
   31 DET = DET*A(I,I)
      DET = DET*ISIGN
        END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR GAUSSP

NO ERRORS

```
      SUBROUTINE INDEX(JJ,I,J,N2)
      JJ= J+((I-1)*(N2-1))/2
        END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR INDEX

NO ERRORS

```
      SUBROUTINE BIGSET (N)
      COMMON SA(13), SB(12), C(78, 79), X(78), T(12, 12), E(12, 12),
     1       A(12, 12), B(12, 12)
      NN=(N*(N+1))/2  $  NNP1= NN+1
      DO 7 I=1, N
      DO 7 J=1, N
    7 B(I,J)= E(I,N)*E(J,N)
      B(N,N)= B(N,N)-1.0
      DO 1 I=1, NN
      DO 1 J=1, NNP1
    1 C(I,J)= 0.0
      N2= 2*N  $  L=0
      DO 2 J=1, N
      JP1= J+1
      DO 2 I=J, N
      L=L+1
      IP1= I+1
      DO 3 K=1, J
      CALL INDEX(JJ,K,J,N2)
    3 C(L,JJ)=C(L,JJ)+A(I,K)
      DO 4 K=JP1, N
      CALL INDEX(JJ,J,K,N2)
    4 C(L,JJ)=C(L,JJ)+A(I,K)
      DO 5 K=1, I
      CALL INDEX(JJ,K,I,N2)
    5 C(L,JJ)= C(L,JJ)+A(J,K)
      DO 6 K=IP1, N
      CALL INDEX(JJ,I,K,N2)
    6 C(L,JJ)=C(L,JJ)+A(J,K)
    2 C(L,NNP1)= B(I,J)
        END
```

3200 FORTRAN DIAGNOSTIC RESULTS - FOR BIGSET

NO ERRORS

```
      SUBROUTINE SOLUTION(N,M,IPARITYI,IPARITYO,ISTOP,IFNPRINT,IFXPRINT)
      COMMON SA(13), SB(12), Z(12),ZP(12), WP(12), W(600), X(5604),
     1        TR(12, 12), E(12, 12), T(12, 12)
      IPARITYI=0
      IPARITYO=0
      BUFFER IN (1,1)(MAXW, MAXW)
   30 GO TO (30, 31, 22, 3), UNITSTF(1)
   31 IWMAX= MAX W
      BUFFER IN(1,1)(W(1),W(IWMAX))
    1 GO TO (1, 2, 22, 3), UNITSTF(1)
    3 IPARITYI=1
      RETURN
    2 MP1= M+1
      MP2= M+2
      II=N
      ISTART=2
      DO 4 I=1, N
      Z(I)=0.0
      DO 4 J=1, I
    4 Z(I)= Z(I)+T(I,J)*W(J)
      X(1)= 0.0
      DO 5 I=1, MP1
      J= MP2-I
    5 X(1)=X(1)+SB(I)*Z(J)
    7 DO 8 I=ISTART, ISTOP
      IPRINT = I-1
    9 II= II+N
      IF(IWMAX .LT. II) 10, 11
   10 IF(II-N .EQ. IWMAX) 12, 13
   12 JJ= 0
      GO TO 14
   13 JJ= II-IWMAX
      KL= II-N
      DO 15 KJ=1, JJ
      KL= KL+1
   15 W(KJ)= W(KL)
   14 BUFFER IN (1,1)(MAX W, MAX W)
   32 GO TO (32, 33, 34, 3) , UNITSTF(1)
   34 IF(IPRINT) 35, 22
   35 IF(IF N PRINT .EQ. 1) 70, 71
   70 PRINT 72, IPRINT
   72 FORMAT(////,I10, 30H POINTS ARE IN THIS DATA BLOCK , /)
   71 IF(IF X PRINT .EQ. 1) 73, 74
   73 PRINT 28, (X(J), J=1, IPRINT)
   74 BUFFER OUT (2,1)(IPRINT, IPRINT)
   62 GO TO (62, 63, 63, 25), UNITSTF(2)
   63 BUFFER OUT(2,1)(X(1), X(IPRINT))
   64 GO TO (64, 22, 22, 25), UNITSTF(2)
   33 IWMAX= MAX W + JJ
      JJ=JJ+1
      BUFFER IN(1,1)(W(JJ),W(IWMAX))
   23 GO TO(23, 24, 22, 3), UNITSTF(1)
   24 II=0
      GO TO 9
   11 JJ= II-N
      DO 16 J=1, N
      JJ=JJ+1
   16 WP(J)= W(JJ)
      DO 17 K=1, N
      ZZP=0.0
```

```
      DO 18 J=1, N
   18 ZZP= ZZP+F(K,J)*Z(J)+TR(K,J)*WP(J)
   17 ZP(K)= ZZP
      XX=0.0
      DO 19 K=1, MP1
      J= MP2-K
   19 XX=XX+SB(K)*ZP(J)
      X(T)= XX
      DO 8 J=1, N
    8 Z(J)= ZP(J)
      IF(IF N PRINT .EQ. 1) 75, 76
   75 PRINT 72, ISTOP
   76 IF(IF X PRINT .EQ. 1) 77, 78
   77 PRINT 28, (X(I), I=1, ISTOP)
   28 FORMAT (6E20.10)
   78 BUFFER OUT(2,1)(ISTOP, ISTOP)
   60 GO TO (60, 61, 61, 25), UNITSTF(2)
   61 BUFFER OUT (2,1)(X(1), X(ISTOP))
   21 GO TO (21, 6, 6, 25), UNITSTF(2)
   25 IPARITYO=1
      RETURN
    6 ISTART=1
      GO TO 7
   22 END FILE 2
      CONTINUE
        END

        3200 FORTRAN DIAGNOSTIC RESULTS - FOR   SOLUTION


UNDEFINED SIMPLE VARIABLES
      MAXW
EQUIP,2=MT
LOAD,56
RUN
```

APPENDIX

In addition to the "white sequence," $\{e^i\}$, $1 \leq i \leq 4266$, discussed in the text of this paper, seven other "white sequences" have been generated by ISLASHO. These eight "white sequences" have been permanently stored on magnetic tape; this tape is labeled Tape B.

The eight "white sequences" (in the order in which they are stored on Tape B) are:

$$\nu_i^{(1)} = \{\theta^i\},\ 1 \leq i \leq 4004;\ \theta = 9.63778431 \approx \log_{10} N$$

$$\nu_i^{(2)} = \{\theta^i\},\ 1 \leq i \leq 4010;\ \theta = 1.27323954 \approx 4/\pi$$

$$\nu_i^{(3)} = \{\theta^i\},\ 1 \leq i \leq 4006;\ \theta = 3.04800610 \approx 0.2(\text{number of cm/ft})$$

$$\nu_i^{(4)} = \{\theta^i\},\ 1 \leq i \leq 4006;\ \theta = 1.46459189 \approx \sqrt[3]{\pi}$$

$$\nu_i^{(5)} = \{\theta^i\},\ 1 \leq i \leq 4006;\ \theta = 4.97149872 \approx \log_{10} \pi$$

$$\nu_i^{(6)} = \{\theta^i\},\ 1 \leq i \leq 4266;\ \theta = 2.71828183 \approx e$$

$$\nu_i^{(7)} = \{\theta^i\},\ 1 \leq i \leq 4008;\ \theta = 2.30258509 \approx 1/N$$

$$\nu_i^{(8)} = \{\theta^i\},\ 1 \leq i \leq 4006;\ \theta = 5.72957795 \approx 0.1(180/\pi).$$

In $\nu_i^{(1)}$ and $\nu_i^{(7)}$, define $N = \log_{10} e$.

$IDEC^{(8)}$

$L^{(8)}$

$IX^{(8)}(1) \cdot \cdot \cdot \cdot IX^{(8)}(L^{(8)})$

end-of-file

end-of-file.

Here IDEC, L, and IX(I) are defined the same as in the section of the text on ISLASHO.

The "white sequence" $\nu_i^{(1)}$ on Tape B can be used as direct input to GAUSSIAN. In order to use $\nu_i^{(s)}$, $2 \leq i \leq 8$, as input to GAUSSIAN, either the specified "white sequence" must first be placed on an auxiliary tape or GAUSSIAN must be modified to skip over previous "white sequences."

The first 4000 points of each of the eight above "white sequences" have been interlaced by the scheme

$$\nu_{8(i-1)+1} = \nu_i^{(1)},\ \nu_{8(i-1)+2} = \nu_i^{(2)},\ \cdot\cdot\cdot\ \nu_{8(i-1)+8} = \nu_i^{(8)},$$

for $1 \leq i \leq 4000$. This 32,000 point "white sequence" has also been stored permanently on a second magnetic tape which is labeled Tape C.

Tape C has the format:

500

$\nu_1 \cdot \cdot \cdot \cdot \nu_{500}$

.

.

.

500

$\nu_{31,501} \cdot \cdot \nu_{32,000}$

end-of-file

end-of-file.

Tape C can be used directly as input to GAUSSIAN.

Tape B has the following format:

500

$v_1^{(1)} \ldots\ldots v_{500}^{(1)}$

.

.

.

500

$v_{3501}^{(1)} \ldots\ldots v_{4000}^{(1)}$

4

$v_{4001}^{(1)} \ldots v_{4004}^{(1)}$

end-of-file

$IDEC^{(1)}$

$L^{(1)}$

$IX^{(1)}(1) \ldots\ldots IX^{(1)}(L)$

end-of-file

.

.

.

500

$v_1^{(8)} \ldots\ldots v_{500}^{(8)}$

.

.

.

500

$v_{3501}^{(8)} \ldots\ldots v_{4000}^{(8)}$

6

$v_{4001}^{(8)} \ldots v_{4006}^{(8)}$

end-of-file

REFERENCES

1 J. N. Franklin, "Numerical Simulation of Stationary and Nonstationary Gaussian Random Processes," SIAM Review, 7, No. 1, 68 (January, 1965).

2 Ibid., pp. 68-80.

3 J. S. Bendat, Principles and Applications of Random Noise Theory (John Wiley and Sons, New York, 1958), pp. 201-6.

4 Ibid., pp. 141, 161-3.

5 R. B. Blackman and J. W. Tukey, The Measurement of Power Spectra (Dover Publications, Inc., New York, 1958), pp. 33-7.

6 Franklin, Op. cit., pp. 68-80.

7 W. B. Davenport and W. L. Root, An Introduction to the Theory of Random Signals and Noise (McGraw-Hill, New York, 1958), pp. 182-3, 225-7.

8 Ibid., pp. 232-4, 375-6.

9 J. N. Franklin, "Deterministic Simulation of Random Processes," Math. Comp., 17, 47-50 (1963).

10 G. E. P. Box and Mervin E. Muller, "A Note on the Generation of Random Normal Deviates," Ann. Math. Statist., 29, 610-1 (1958).

11 J. N. Franklin, "The Covariance Matrix of a Continuous Autoregressive Vector Time Series," Ann. Math. Statist., 34, 1259-64 (1963).

12 F. B. Hildebrand, Introduction to Numerical Analysis (McGraw-Hill, New York, 1956), pp. 486-9.

13 L. Fox, An Introduction to Numerical Linear Algebra (Oxford University Press, New York, 1965), pp. 60-91.

14 J. N. Franklin, "Numerical Simulation of Stationary and Nonstationary Gaussian Random Processes," SIAM Review, I, No. 1, 70 (January, 1965).

15 Defense Research Laboratory, The University of Texas, Acoustical Report No. 252 resulting from research under U. S. Naval Oceanographic Office Contract N62306-1358, 8 December 1965.

[16] Defense Research Laboratory, The University of Texas, Annual Report No. 1 on Contract DA 28-043 AMC-00316(E), 1 July 1964-30 June 1965, pp. 27, 31.

[17] *Ibid.*, p. 23.

[18] *Ibid.*, p. 27.

BIBLIOGRAPHY

Bendat, J. S., Principles and Applications of Random Noise Theory (John Wiley and Sons, New York, 1958).

Blackman, R. B. and J. W. Tukey, The Measurement of Power Spectra (Dover Publications, Inc., New York, 1958).

Box, G. E. P. and Mervin E. Muller, "A Note on the Generation of Random Normal Deviates," Ann. Math. Statist. 29, 610-1(1958).

Davenport, W. B. and W. L. Root, An Introduction to the Theory of Random Signals and Noise (McGraw-Hill, New York, 1958).

Ellis, G. E. and R. L. Boston, "Program SPECT: Power Spectral Analysis of a Random Process Using the CDC 1604 Computer," Defense Research Laboratory, The University of Texas, Austin, Texas, Acoustical Report A-252 (December, 1965).

Fox, L., An Introduction to Numerical Linear Algebra (Oxford University Press, New York, 1965).

Franklin, J. N., "The Covariance Matrix of a Continuous Autoregressive Vector Time Series," Ann. Math. Statist. 34, 1259-64 (1963).

Franklin, J. N., "Deterministic Simulation of Random Processes," Math. Comp. 17, 28-59 (1963).

Franklin, J. N., "Numerical Simulation of Stationary and Nonstationary Gaussian Random Processes," SIAM Review 1, No. 1, 68-80 (January, 1965).

Hildebrand, F. B., Introduction to Numerical Analysis (McGraw-Hill, New York, 1956).

Richie, W. C. and D. R. Chick, "Report on Research to Investigate Atmospheric Infrasound," Defense Research Laboratory, The University of Texas, Austin, Texas, Acoustical Report A-249 (November, 1965).

27 October 1966

DISTRIBUTION LIST FOR DRL-A-258
UNDER CONTRACT NObsr-93175

Copy No.	
1 - 4	Commander, Naval Ship Systems Command Department of the Navy Washington, D. C. 20360 Attn: Mr. C. D. Smith (Code 1622C)
5	Commanding Officer U. S. Navy Mine Defense Laboratory Panama City, Florida 32402 Attn: Mr. Carl M. Bennett
6	Commanding Officer U. S. Naval Oceanographic Office Suitland, Maryland 20390 Attn: Mr. R. C. Guthrie (Code 7240) Oceanographic Analysis Division Marine Sciences Department
7	Commanding Officer U. S. Navy Electronics Laboratory San Diego, California 92152 Attn: Code 3180
8	Office of Naval Research Resident Representative The University of Texas 2507 Main Building Austin, Texas 78712
9	Acoustics Division, DRL/UT
10	Signal Physics Branch, DRL/UT
11	K. J. Diercks, DRL/UT
12	G. E. Ellis, DRL/UT
13	D. W. Evertson, DRL/UT
14	K. W. Harvel, DRL/UT
15	S. P. Pitt, DRL/UT
16 - 17	Library, DRL/UT